DESIGNING HIGHLY EFFICIENT AND RELIABLE SECURE TWO-ROUND MULTI-SIGNATURE SCHEME

Kaoru Takemure

The University of Electro-Communications

Graduate School of
Informatics and Engineering

A Dissertation Submitted for
Doctor of Philosophy in Engineering

March, 2024

DESIGNING HIGHLY EFFICIENT AND RELIABLE SECURE TWO-ROUND MULTI-SIGNATURE SCHEME

Supervisory Committee

Chairperson:	Professor	Yasutada Oohama
Member:	Professor	Mitsugu Iwamoto
Member:	Professor	Kazuki Yoneyama
Member:	Associate Professor	Tomohiro Ogawa
Member:	Associate Professor	Hideki Yagi
Member:	Associate Professor	Bagus Santoso

和文概要

電子署名方式は、現実世界において印鑑のような役割を担う暗号方式である。 具体的には、署名者は秘密鍵を用いて文書に対して署名を生成し、検証者は公開鍵を用いて署名検証を行う。 近年、ブロックチェーンを利用した暗号資産等のアプリケーションにおいては、それぞれ自身の秘密鍵を持つ複数の署名者で共通の文書に対して署名を生成するという状況が考えられる。 通常の電子署名方式を利用する素朴な方法として、複数の署名者により生成される複数の署名を１つの署名と見なし、検証者はすべての署名が正当であれば合格とする方法が考えられる。 しかしながら、この方法では署名のサイズが署名者数に線形に依存して増大してしまう。 これは、暗号資産の抱える課題の１つであるスケーラビリティ問題を悪化させる。

上記の問題を解決可能な署名方式の１つとして、多重署名方式がある。 多重署名方式では、複数の署名者が各々の秘密鍵と公開鍵を持ち、１つの文書を共有した署名者全員で署名生成プロトコルを実行することで、署名者数に依存しない署名長の署名を生成する。 署名は、すべての署名者の公開鍵により検証可能である。 したがって、多重署名方式を用いることで、署名長を大幅に悪化させることなく、複数の署名者で共通の文書に署名を発行することができる。 多重署名方式の効率性の評価軸として、署名長だけでなく、署名生成プロトコルの通信回数(ラウンド)や通信量も重要な指標となる。 本研究では、特に離散対数ベースの多重署名方式に着目する。 Bitcoin等では離散対数ベースの署名方式が用いられているため、離散対数ベースの多重署名方式は高い互換性を持つ。 また、ライブラリも充実しており、実装は比較的容易である。 一般に非対話型の離散対数ベースの多重署名方式の構成は困難であることが知られているため、 署名生成プロトコルの方式は2ラウンドが最適と考えられている。

離散対数ベースの多重署名方式において、現在2ラウンドの署名生成プロトコルを達成する方式は複数提案されている。 既存の2ラウンド方式らは、帰着ロスの大きい方式と小さい方式に分類できる。 帰着ロスとは、安全性証明より導ける、方式を破る困難性と計算問題の困難性との差を示す指標である。 信頼性の高い安全性を達成するためには、実装す

る際に用いるパラメタは帰着ロスを考慮して設定される必要がある。 一般に帰着ロスが大きくなるほどより大きなパラメタが必要となる。 大きな帰着ロスを持つ既存方式は、信頼性の高い安全性を保証するためには非常に大きなパラメタを必要とし効率性は悪い。 また、これらの方式は、標準的な安全性レベルである128ビット安全性を保証可能な標準化楕円曲線を持たない。 帰着ロスが小さい既存方式は、小さいパラメタを用いることができるため高効率であり、また128ビット安全性を保証可能な標準化楕円曲線を持つ。 一方で、これらの方式は、すべての離散対数ベースの既存方式で用いられているハッシュ関数に関する理想的な仮定であるRandom Oracle Model (ROM)に加え、さらに攻撃者の演算に制限を設けるAlgebraic Group Model (AGM)という理想的な仮定も用いている。 現在、AGMを使わずにROMのみを用いることで、128ビット安全性を保証可能な標準化楕円曲線を持つ程の小さい帰着ロスを達成する効率的な多重署名方式、すなわち高い効率性と信頼性の高い安全性の両方を達成する方式は知られていない。

本論文では、そのような高効率と高信頼の安全性の両方を達成する初めての2ラウンド多重署名方式を提案する。 本研究では、安全性の根拠となる計算問題が判定問題であるような2ラウンド方式を構成する新たな手法を考案することで提案方式を構成する。 具体的には、この手法は、安全性の根拠として探索問題を用いるある既存2ラウンド方式で用いられているテクニックを基に考案される。 本論文では提案方式を2つ提案する。1つ目の方式は既存方式より微妙に弱い安全性のみを達成する方式である。 2つ目の方式は1つ目の改善方式であり、効率性の悪化や仮定を追加することなく既存方式と同等の安全性を達成可能な方式である。 提案方式は、AGM は用いずに、判定Diffie-Hellman(DDH)仮定とROMの下で安全性が証明される。 DDH 問題は、実用的な世界では離散対数問題と同程度の困難性を持つと考えられている標準的な計算問題の 1 つである。 提案方式が小さい帰着ロスを達成したことにより、この方式は128ビット安全性を保証可能な標準化曲線を持つこととなった。 提案方式はAGMを用いずにこれを達成する初めての方式である。 提案方式は、AGMを用いない方式の中で最も小さい署名長と通信量を達成している。 また、提案方式について実装を行い、署名生成および署名検証の計算時間を測定し、実用的な計算時間であることを確認した。

本研究の結果は、離散対数ベースの多重署名方式において、安全性の信頼性、効率性、仮定の強さ間における新たなトレードオフを示唆している。 これは、よりユーザの要求に適したアプリケーションの実現可能性を高めることが期待される。 本論文の提案方式は、安全性の信頼性と方式の効率性の両方を要求するユーザに対し、最も適した方式であるといえる。

Abstract

Digital signatures are one of the fundamental cryptographic primitives. In digital signatures, a signer generates a signature on a message to prove the validity of a message by using a secret key. The signature is verified by the corresponding public key. In blockchain-based applications, e.g., cryptocurrencies, we consider a situation where multiple signers generate signatures on the same message m. The naive approach to achieve this is as follows. Multiple signers generate signatures on m by using their own secret key. The set of signatures is regarded as a signature on m. A verifier accepts a message if all signatures are valid. However, this approach makes signature schemes inefficient since the size of the signature grows linearly with the number of keys. This exacerbates an issue in cryptocurrencies, namely, the problem of scalability. Thus more advanced cryptosystems are needed.

Multi-signatures are one of the cryptosystems that solve this issue. In a multi-signature scheme, for a single common message m, multiple signers cooperatively generate a multi-signature σ, which is a combination of multiple individual signatures on the message m where each is created by each party using its secret key. The multi-signature σ is verified by all public keys involved in the signing protocol. An essential property of the multi-signatures is that its size is kept constant independently of the number of signers. Thus using a multi-signature scheme allows us to generate the constant size signature on m by multiple signers without making the issue in cryptocurrencies worse. In addition to the signature size, the number of communication rounds and the communication complexity of the signing protocol are also important factors of efficiency. In this research, we focus on the discrete logarithm (DL)-based multi-signature scheme. Since the DL-based signature schemes are used in current cryptocurrencies, e.g., Bitcoin, they are highly compatible. Also, they are easy to implement due to great library support. Due to the widely held belief that it is hard to construct a non-interactive DL-based multi-signature scheme, a two-round signing protocol is regarded as optimal.

To date, several DL-based two-round multi-signature schemes have been

proposed. We can classify the existing schemes into two types, schemes with non-constant reduction losses (non-tight security) or with constant reduction losses (tight security). The reduction loss expresses the gap between the hardness of breaking the security of the cryptographic primitive and that of solving the computational problem on which the security of the primitive is based. When we guarantee highly reliable security, we need to implement schemes under provable secure parameters which are derived by considering the reduction loss. The existing schemes with non-tight security have large reduction losses which does not allow us to ensure the standard security level, e.g., 128-bit security, under standardized elliptic curves (EC), e.g., NIST Prime Curves. Since these schemes require large parameters, these schemes are inefficient under provable secure parameters. While the other existing schemes with tight security can ensure 128-bit security and short signature size under provable secure parameters, they require not only the Random Oracle Model (ROM) but also the Algebraic Group Model (AGM) which are idealized models of hash functions and computation, respectively. Note that all existing DL-based schemes use the ROM to prove the security of them. At present, there is no two-round scheme achieving a small reduction loss, which can ensure 128-bit security under the standardized EC, without using the AGM, namely, a scheme that achieves both high efficiency and reliable security.

In this thesis, we propose new two-round multi-signature schemes that achieve high efficiency and reliable security. We construct our scheme by devising a new approach to construct a two-round scheme whose security relies on a decisional problem. This approach is devised from a technique employed in an existing scheme, the security of which is proven under a search problem. We propose two schemes. The first scheme only achieves slightly weak security compared with related schemes. The second scheme is an improvement of the first scheme. This is proven secure as same as existing schemes without compromising the efficiency and adding assumptions. The security of our schemes is proven under the decisional Diffie-Hellman (DDH) assumption in the random oracle model (ROM). In practice, the DDH problem is considered to be about as hard as the DL problem. Moreover, due to achieving the small reduction loss, our scheme can use the standardized elliptic curve (EC) to ensure 128-bit security, in practice. The use of a standardized EC guarantees reliable implementations. Our scheme is the first scheme that achieves it without using the AGM. Also, our signature size and communication complexity are the smallest among the schemes without using the AGM. In addition, 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.

Our proposed scheme shows new trade-offs between reliability of security, efficiency, and strength of underlying assumptions. This gives us a new candidate for multi-signatures that meets the user's requirements. This enhances the feasibility of cryptocurrencies and blockchain-based applications, making them better suited to meet user demands. Indeed, our scheme is desirable for users who do not want to compromise on the reliability of the security, efficiency, and strength of assumptions.

Publication Related to This Thesis

1. Kaoru Takemure, Yusuke Sakai, Bagus Santoso, Goichiro Hanaoka, Kazuo Ohta, “More Efficient Two-Round Multi-Signature Scheme with Provably Secure Parameters for Standardized Elliptic Curve”, IEICE Transaction on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E107-A, No.7, pp.-, Jul. 2024. (to appear, total number of pages: 24)

Acknowledgements

I would like to deeply thank my main supervisor Professor Yasutada Oohama for helpful suggestions regarding my research and great support. I would like to express my appreciation to Associate Professor Bagus Santoso. The fascination with the research of cryptography was first taught to me by him. Since I joined his lab in my fourth year of undergraduate studies, I have had fruitful discussions and received great support and advice from him, not limited to research. I also would like to thank Associate Professor Hideki Yagi for meaningful suggestions and comments in the joint seminar in the laboratory. I would like to express my appreciation to all of them for their helpful and significant comments on my thesis and presentation.

I would like to gratefully thank the supervisory committee members, Professor Mitsugu Iwamoto, Professor Kazuki Yoneyama, and Associate Professor Tomohiro Ogawa. Their comments and suggestions were insightful and provided valuable guidance for enhancing my thesis and presentation.

I would like to gratitude to the members of Cyber Physical Security Research Center at National Institute of Advanced Industrial Science and Technology. I would like to deeply thank Goichiro Hanaoka and Takahiro Matsuda for their thoughtful support and advice, without limited to research. I would like to thank Kazuo Ohta and Yusuke Sakai for fruitful discussions and insightful suggestions in research, including that of multi-signatures of this thesis. I would like to thank Shota Yamada and Shuichi Katsumata for the productive discussions and precious experiences of recent joint work. I also would like to thank other members for sharing the latest research results and interesting research.

I would like to thank all members of Santoso Laboratory for the discussions, delightful chats, and enjoyable time. I have enjoyed my research life in this laboratory thanks to them. Thank you, Yuto Kou, Taichi Yaguchi, Akitaka Yokota, Arga Dhahana Pramudianto, Tihang Wijaya, Daigo Kuroki, Takumi Yada, Toma Uetsu, Subaru Fuji, Riku Yoneya, Koshiro Yamashita, Jin Yang, and Lukas Palmqvist.

Finally, I would like to thank my family for their financial and mental

support.

Contents

Chapter 1

Introduction

1.1 Backgrounds

1.1.1 Digital Signatures

Digital signatures [DH76, RSA78, GMR88] are cryptosystems that ensure the validity of digital data. In the physical world, we prove the authenticity of a document through a signature or a stamp. Digital signatures make this possible in the digital realm. Along with public key encryptions, digital signatures are one of the most fundamental cryptographic systems. In practice, digital signature schemes are used in numerous systems.

A digital signature scheme is defined by three efficient algorithms, the key generation algorithm, the signing algorithm, and the verification algorithm. The key generation algorithm provides the public key and secret key. The public key is publicly opened and the secret key is kept secret by the signer. To authenticate a message, the signer generates a signature by using the signing algorithm and the secret key. Anyone having the public key can verify the signature by using the verification algorithm.

The security of digital signatures guarantees that anyone who does not know the secret key cannot generate a forgery. This security notion is called unforgeability. The unforgeability of most signature schemes is proven under certain computational assumptions. In other words, we prove that breaking the unforgeability of a scheme is about as hard as solving a certain computational problem. To prove this statement, we usually show that the contraposition of the statement holds. Namely, we show that there exists an efficient algorithm solving a certain problem if there exists an efficient adversary breaking the security of a scheme. Note that this way is a common practice in provable security and is not limited to digital signatures.

The notion of digital signatures was introduced by Diffie, Hellman, and

Merkle [DH76, Mer78, MH78]. Rivest, Shamir, and Adleman constructed the public-key encryption scheme, a.k.a. RSA, and also proposed the first digital signature scheme based on it [RSA78]. Goldwasser, Micali, and Rivest defined the security classes of digital signatures. Currently, the typical security class that needs to be guaranteed is existential unforgeability under a chosen message attack (EUF-CMA). This security notion ensures that any adversary cannot forge a signature on any message even though it can obtain valid signatures on messages that are adaptively chosen by it and are not the same as the message to be forged. They also constructed the first digital signature scheme that achieves EUF-CMA.

Bellare and Rogaway introduced the notion of the random oracle model (ROM) and constructed efficient signature schemes from the trapdoor one-way function, which is called the full-domain hash signatures [BR96]. These schemes were proven EUF-CMA under the ROM. While the ROM is an idealized model for the hash function, it allows us to obtain efficient schemes. Fiat and Shamir proposed a way to transform from an interactive identification to a digital signature, which is called the Fiat-Shamir transform [FS87]. The Schnorr signature scheme is one of the famous digital signature schemes based on the discrete logarithm problem constructed by this transformation [Sch90]. No formal security proofs of such signature schemes were given at the time. Pointcheval and Stern proved that the Schnorr signature scheme is EUF-CMA under the ROM by using the rewinding technique [PS96]. Boneh et al. proposed a practical signature scheme based on pairing [BLS01]. Numerous signature schemes that are proven secure under the ROM are constructed from various assumptions so far.

There are some practical digital signature schemes that are secure in the standard model, in which the ROM is not used. The signature schemes [GHR99, CS99] were proven secure in the standard model under the strong RSA assumption. It is known that digital signature schemes can be obtained from identity-based encryption schemes (IBE) [BF01, SOK00]. In this manner, signature schemes based on the pairing [BB04, Wat05, Gen06] were proposed.

1.1.2 Multi-Signatures

In blockchains and cryptocurrencies, we often face a situation where multiple signers produce signatures on the same message M. The naive way is that each signer generates a signature on M by using his secret key. The signature on M consists of all signers' signatures. Then, a verifier accepts the signature if all signatures are valid. This is known as Multi-Sig in the context of cryptocurrencies.

Multi-Sig is one of the ways to improve the security of cryptocurrencies, e.g., Bitcoin [Nak08]. The theft of funds is a serious problem for them. The cause of this is the compromise of secret keys of digital signatures. When a sender spends funds, it produces a signature on a transaction by using its secret key to certify the validity of a transaction. However, if the secret key is compromised, anyone can generate a valid signature on a fake transaction and consequently steal funds. In fact, many incidents have occurred so far due to secret key compromises. Multi-Sig is one of the ways to prevent secret key compromise. In a nutshell, a sender keeps multiple secret keys in a distributed manner and produces multiple signatures associated with these keys (or a part of these keys) to spend funds. Due to this countermeasure, even if a part of the secret keys are leaked, we can prevent the theft of funds. The statistical study [dAS20a, dAS20b] on the blockchain-based cryptocurrency Ethereum's main chain up to block 1,1500,000 (mined on Dec 22, 2020) shows that 12.5% of all wallets are actually Multi-Sig Wallets.

However, unfortunately, this approach exacerbates another problem facing cryptocurrencies, i.e., the problem of scalability. Specifically, if we develop Multi-Sig in a system, the size of the signature to be stored in a block increases linearly with the number of signers. However, there is a limit to the amount of data that can be stored in one block. Thus, increasing the size of signatures included in transactions causes delays in the transaction processing and increases the transaction fees. Therefore, the countermeasures without increasing the signature size are desirable.

Fortunately, we have some solutions, one of which is *multi-signatures* [IN83]. In the multi-signatures, for a single common message M, multiple signers cooperatively generate a signature $\widetilde{\sigma}$, known as a multi-signature, which is basically a combination of multiple individual signatures $(\sigma_i)_i$ on M where each is created by each signer using its own secret key. The unforgeability of multi-signature schemes guarantees that any efficient adversary cannot forge, even if it corrupts all signers except for one signer. The essential and worthwhile property of multi-signatures is that the size of multi-signatures is independent of the number of signers. Therefore, using a multi-signature scheme instead of the naive way of Multi-Sig improves the security against the theft of funds without harming the scalability. Multi-signature schemes based on several hardness problems have been proposed so far, e.g., the discrete logarithm (DL)-based schemes [MOR01, BN06, MPSW19, DEF$^+$19, NRSW20, NRS21, AB21, BD21, LK22, TZ23, PW23], pairing-based schemes [Bol03, BGOY07, LBG09, LOS$^+$06, RY07], lattice-based schemes [ES16, MJ19, FH20, DOTT21, BTT22, Che23a], and so on.

Remark 1. *We can use the threshold signatures [Des90, DF90] as another solution. In T-out-of-N threshold signatures, a secret key is distributed to N*

signers and a signature on a message is generated by any set of $T \leq N$ signers. Multi-signatures can be regarded as a special case of threshold signatures, i.e., N-out-of-N threshold signatures. In the threshold signatures, all signers generate a public key and secret key shares by executing the distributed key generation (DKG). While we can easily execute the DKG if there is a trusted third party, all signers execute the DKG protocol by interacting with each other without such a trusted one. The threshold signatures are verified by one public key. In contrast, all signers generate public and secret keys by themselves in the multi-signatures, and the multi-signatures are verified by multiple public keys.

Discrete-Logarithm-Based Multi-Signatures. In this research, we focus on the (pairing-free) Discrete Logarithm-based (DL-based) multi-signature schemes, which are well-studied by a lot of literature referred to above. The DL-based scheme can be implemented under the elliptic curves used to implement standard digital signature schemes, e.g., the ECDSA [Nat13] and the Schnorr signature scheme [Sch90], used in some cryptocurrencies, e.g., Bitcoin. In such applications, DL-based multi-signature schemes have high compatibility. While the use of pairing, which has a special algebraic structure, i.e., the bilinear map, allows us to construct aggregatable signature schemes including multi-signature schemes (see Section 1.5), pairing-friendly elliptic curves are not supported by highly-verified standard cryptographic libraries, e.g., NSS and BoringSSL, as mentioned in [CKM+23]. This suggests that focusing on the pairing-free DL-based scheme which can be implemented under the standard pairing-free elliptic curves supported by such libraries is worthwhile. Now we briefly review the DL-based multi-signature schemes. For more detailed previous research of DL-based multi-signature schemes, see Chapter 3.

The multi-signature schemes proposed in early literature [IN83, LHL95, Lan96, MH96, OO93, OO99, Har94] are not secure against the rogue-key attack, in which an adversary maliciously generates cosigners' public keys to forge. A naive way to prevent this attack is to attach the certification of knowledge of a secret key to a public key. Another way is to execute a trusted key generation protocol [MOR01]. However, these ways make the scheme inefficient and are not suitable for some applications.

Bellare and Neven proposed the first secure three-round multi-signature scheme without a trusted key setup [BN06]. Such an unrestricted key setup model is called the plain public key (PPK) model. Here 'three-round' means that the signing protocol has three-round communications between signers. Also, Maxwell et al. [MPSW19] introduced a notion of key aggregation, which provides the compression of public keys and efficient verification, in the context of the application of cryptocurrencies. Due to these works, the primary desirable features of the multi-signatures are the security in the PPK model, the two-round signing protocol, and the support of the key aggregation.

Although some two-round multi-signature schemes [BCJ08, MWLD10, STV+16, MPSW18] were proposed after the three-round scheme [BN06] was proposed, Drijvers et al. [DEF+19] suggested the vulnerability of them by demonstrating attacks, which were improved by [BLL+21]. They also proposed the first secure two-round multi-signature scheme. To date, several schemes achieving the three desirable properties described above have been proposed [NRSW20, NRS21, AB21, BD21, LK22, PW23, TZ23].

1.2 Concrete Security

1.2.1 Concrete Security and Tightness

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 the working factors $\mathsf{WF}_A = t_A/\epsilon_A$ and $\mathsf{WF}_P = t_P/\epsilon_P$, 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. Intuitively, the working factors WF_A and WF_P express the expected time required to break the cryptosystem and solve the computational problem, respectively. A typical relation between WF_P and WF_A derived from the security proof is as follows: $\mathsf{WF}_A \geq \mathsf{WF}_P/\Phi$, where $\Phi \geq 1$ is often referred to as the *reduction loss*. In concrete security, the parameters of the cryptosystem are provided from this relation. We call such parameters as *provable secure parameters*.

Provable secure parameters are significant from the point of view of reliable security. The gap, i.e., the reduction loss Φ, suggests the existence of a potential attack against the cryptosystem. This means that the parameters without considering Φ may be not sufficient to ensure the security level we desire. However, 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, a reduction loss Φ was often disregarded. Then, this disregard sometimes makes schemes vulnerable. Indeed, there are some examples of this vulnerability [CMS12, KZ20]. In [KZ20], Kales and Zaverucha demonstrate an attack on the MQDSS signature scheme [SSH11]. Their attack exploits the fact that the parameter of MQDSS was derived without considering Φ. Therefore, it is important to derive the parameters by considering Φ based on the security proof, and thus we should implement cryptosystems with provable secure parameters for reliable security.

In general, 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*. In contrast, we say that the security proof is non-tight if Φ depends on those parameters. The tight security proof states that breaking the cryptosystem is as hard as solving a certain computational problem. There are numerous works that study the tight security of many cryptographic primitives from theoretical and practical aspects, e.g., (identity-based) public-key encryption [BBM00, HJ12, LJYP14, LPJY15, HKS15, AHY15, BJLS16, GHKW16, GCD+16, Hof17], signatures [GJKW07, Sch11, HJ12, AFLT12, BKKP15, BJLS16, BL16, KMP16, DGJL21, PW22], multi-signatures [BN05, WSQL08, PP16, Yan18, FH19, FH21, KSH23, PW23], etc. Hence, research directed toward achieving tight security is meaningful both in practice and theory.

1.2.2 Benefits for Efficiency and Reliability of Implementation

Under considering concrete security, tight security is preferable from the aspect of efficiency. To ensure 128-bit security, which guarantees $\mathsf{WF}_A \geq 2^{128}$, the cryptosystem with tight security can be implemented under the provable secure parameters whose size ensures $\mathsf{WF}_P = 2^{128}$. In contrast, if Φ is large, we need to ensure that WF_P is sufficiently large so that the derived lower bound of WF_A, i.e., WF_P/Φ, is not too small to have a practical meaning. Usually, the only way to make WF_P larger is by setting larger parameters, which means higher costs for implementation in practice. Thus the smaller Φ is desired in practice.

We now clearly explain the effect of the reduction loss on efficiency by demonstrating how provable secure parameters are derived. Let us consider a scheme that is proven secure under the DL assumption. The known fastest algorithm for solving the DL problem is Pollard's ρ algorithm [Pol78], which requires $O(\sqrt{p})$ time where p is the prime order of the underlying group. Then, we set a 256-bit prime integer to p to obtain $\mathsf{WF}_P \geq 2^{128}$. Typically, the parameters guaranteeing 128-bit security are set so that $\mathsf{WF}_A \geq 2^{128}$. Now we show how to derive provable secure parameters. When $\Phi = 1$ holds, we can set a 256-bit prime integer to p to ensure 128-bit security since $\mathsf{WF}_A \geq \sqrt{2^{256}} \geq 2^{128}$ holds. This means that we can use the group of 256-bit prime order. When $\Phi = 2^{60}$ holds, we can use the group of a 376-bit prime order since $\mathsf{WF}_A \geq \sqrt{2^{376}}/2^{60} \geq 2^{128}$ holds. When Φ is 2^{160}, a scheme requires a group of 576-bit prime order, since $\mathsf{WF}_A \geq \sqrt{2^{576}}/2^{80} \geq 2^{128}$ holds. This demonstration suggests that a scheme with a smaller Φ can provide security as reliable as one with a large reduction loss but using smaller parameters.

A smaller reduction loss also makes implementation reliable due to the use of standardized tools. For example, for the DL-based schemes with tight security, an elliptic curve (EC) with a 256-bit prime order ensures to guarantee 128-bit security, as shown in the demonstration above. Then, we can use the standardized ECs, e.g., NIST P-256 [CMR+23] and Secp256k1 [Bro10], for 128-bit security. 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. Specifically, in the second example considered in the demonstration above, we are able to use the standardized ECs, e.g., NIST P-384 and P-521 [CMR+23], instead of that with a 256-bit order. Whereas, it is difficult for a scheme with a large reduction loss to use the standardized cryptographic tools. Indeed, there is no standardized EC for 128-bit security in the last case of the above demonstration. For such a scheme with very large Φ, we need to design a new desirable EC. This makes the implementation difficult and less reliable. In this thesis, for the DL-based scheme, if Φ is not constant but sufficiently small to ensure the existence of a standardized EC for 128-bit security, we say that the reduction loss is small. If not so, we say that the reduction loss is large.

1.3 Motivation

Here, we review the existing DL-based two-round multi-signature schemes in terms of the tightness of a reduction. We note that the security of all related schemes is proven in the random oracle model (ROM) [BR93].

Most of them can be categorized into two types: The first type is the schemes with non-tight security (namely, having a large reduction loss) and the second type is the schemes with a tight security. The schemes of the first type include MuSig-DN [NRSW20], MuSig2-1 [NRS21][1], HBMS [BD21], TZ [TZ23], and mBCJ-PPK [DEF+19][2], while the schemes of the second type include MuSig2-2 [NRS21], DWMS [AB21], HBMS-AGM [BD21][3], LK [LK22].

[1] MuSig2 is proven secure both with and without the use of the AGM. We call the former and latter as MuSig2-2 and MuSig2-1, respectively.

[2] The original of mBCJ is proven secure in a restricted key setup model. In Section 3.4.1, we present a variant of it that is secure in the PPK model. We call this variant scheme mBCJ-PPK.

[3] HBMS is proven secure both with and without the use of the AGM. We especially call the former as HBMS-AGM.

Note that for MuSig2-1, MuSig2-2, TZ, and DWMS, the first round of the signing protocol can be done as pre-processing.

When taking tightness into consideration, even for 128-bit security, the first-type schemes require elliptic curves (EC) with a very large order. In order to provably ensure 128-bit security, MuSig-DN, MuSig2-1, HBMS, TZ, and mBCJ-PPK, respectively, require 740-bit, 750-bit, 986-bit, 742-bit, and 574-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 problem (or other search problem), the reduction performs the *rewinding* of the adversary. This is similar to the security proof of the Schnorr signature scheme, on which these schemes are constructed based. Moreover, for schemes with key aggregation, the number of rewindings has to be increased. Thus, these schemes have larger reduction losses compared to the Schnorr signature scheme.

The schemes of the second type achieve tight security by using not only the ROM but also the Algebraic Group Model (AGM) [FKL18], which is an idealized model of computation. The schemes allow us to use an EC of a small order, e.g., 256-bit. Then, we can use NIST P-256 or Secp256k1 to implement such schemes, resulting in high efficiency.

From the perspective of minimizing the usage of idealized assumptions, we can consider a scheme that is proven secure by relying on only the ROM as more desirable than one that is requiring both the ROM and the AGM to prove the security. It is known that the Schnorr signature scheme cannot be proven secure in the standard model [PV05]. This implies that the ROM is necessary to prove the security of DL-based multi-signature schemes based on the Schnorr signature scheme. This fact allows us to consider that a scheme whose security does not rely on the AGM is desirable in terms of minimizing the use of ideal models.

Notes of Validity of the ROM and the AGM. Bellare and Rogaway introduced the ROM in 1993 [BR93]. The difference between concrete hash functions and the random oracle has been scrutinized for thirty years. Canetti et al. constructed artificial schemes that are secure in the ROM but insecure with any implementation of the ROM [CGH98]. Moreover, much cryptanalytic literature investigates and analyzes the difference between a concrete hash function from a random oracle [Nie02, GK03, BBP04, CGH04, MRH04, DOP05, AM09, KN10, LMR+09, GP10, RSS11, KM15, Zha22a]. These lines of research provide a more fine-grained understanding of how far (or near) concrete hash functions are from a random oracle.

The AGM is an assumption introduced by Fuchsbauer et al. in 2018.

In this idealized model, when an adversary outputs a group element, it is required to output the linear representation of it relative to all group elements received so far. The gap between this assumption and the real world is investigated by some recent research [KP19, AHK20, Zha22b, ZZK22]. In [KP19], Kastner and Pan instantiate the AGM from the knowledge of exponent assumption [Dam92, BP04, WS07], which is unfalsifiable. After that, Agrikola et al. [AHK20] instantiate the AGM from a falsifiable but strong computational assumption, which is the existence of subexponentially strong indistinguishability obfuscation. Zhandry showed the one-time message authentication code that is secure in the AGM but insecure in the standard model [Zha22b].

The difference between AGM and ROM is the duration of research to provide an understanding of how far the models are from real-world implementations. As mentioned above, while the gap for the ROM has been investigated for three decades, that for the AGM has been studied for half of a decade. The AGM is expected to be better supported by further research.

As concurrent and independent of our result, Pan and Wagner proposed two two-round multi-signature schemes that can guarantee 128-bit security under standardized ECs. The first scheme PW-1 achieves tight security but does not support key aggregation. The second scheme PW-2 has a small reduction loss, e.g., $O(Q_S)$ where Q_S is the number of the signing queries.[4] The second scheme is more efficient than the first one and supports key aggregation. Both are proven secure under the decisional Diffie-Hellman (DDH) assumption in the ROM.

However, under provably secure parameters, the two schemes do not improve the signature size and the communication complexity over the existing non-tight secure schemes even though those achieve tight security or a small reduction loss. Indeed, as shown in Table 1.1, the signature size of PW-1 is largest among the existing schemes without using AGM and the size of PW-2 is larger than MuSig-DN and MuSig2-1.

On this context, we have the following question.

> *Can we construct a two-round multi-signature scheme that achieves both (i) reliable security and (ii) high efficiency while minimizing the use of idealized models?*
> *More specifically, can we construct a two-round signature scheme with a small reduction loss without using the AGM while achieving a short signature size?*

[4] In [GHKP18], Q_S for the multi-signatures is set about 2^{30}. We note that the signers can control the number of the signing queries by regenerating keys.

1.4 Our Contribution

In this thesis, we provide a positive answer to the above question by proposing a two-round multi-signature scheme HBMSDDH-1, which achieves both (i) and (ii) mentioned above. We prove that HBMSDDH-1 is secure under the DDH assumption in the ROM. HBMSDDH-1 guarantees 128-bit security under a standardized EC, e.g., NIST P-384. The signature size and the communication complexity under provable secure parameters are the most efficient among the existing two-round schemes without using the AGM. Moreover, our scheme is proven secure in the PPK model and supports key aggregation, which are desirable property for multi-signatures. Note that HBMSDDH-1 only achieves the slightly weak unforgeability. We also propose a variant HBMSDDH-2 which achieves the standard unforgeability without compromising the efficiency.

Our schemes have reduction losses $O(Q_S)$ where Q_S is the number of signing queries of an adversary. As the result of the estimation of provable secure parameters under the setting $Q_S = 2^{30}$, it only needs an EC with at least 321-bit order to ensure 128-bit security.[5] Therefore, the curve P-384 is sufficient. Under P-384, the signature sizes and the communication complexity per one signer of both schemes are 1152 bits and 1538 bits, respectively. These values achieve the shortest size among the existing schemes without using the AGM. Below, we compare our scheme with the existing scheme in detail. Note that we focus on HBMSDDH-1 although it only achieves slightly weak security because HBMSDDH-2 is as efficient as this scheme.

Firstly, we compare our scheme with the existing non-tight schemes without using the AGM, e.g., MuSig-DN, MuSig2-1, HBMS, TZ, and mBCJ-PPK. Our signature size is reduced by more than 22%, 23%, 60%, 45%, and 49%, respectively. Compared to MuSig2-1, HBMS, TZ, and mBCJ-PPK, our total communication complexity is reduced by more than 59%, 48%, 65%, and 46%, respectively. Note that the first round for MuSig2-1 and TZ can be executed as pre-processing. For these two schemes, the communication complexity in online communication are 750 and 1484, respectively, which are smaller than ours. While the public key of our scheme consists of two group elements, those of those schemes consist of only one group element. However, the public key size of our scheme under P-384 is 770 bits. This size is almost the same as theirs except for mBCJ-PPK. While the key size of mBCJ-PPK is smaller than ours, it does not support key aggregation. Therefore, we conclude that our scheme is more efficient compared to the existing schemes whose security does not rely on the AGM when we consider concrete secu-

[5] We explain the way to estimate provable secure parameters in Section 5.1.1.

rity and ignore the online-offline paradigm. Considering the online-offline paradigm, the online communication complexity of MuSig2-1 and TZ are more efficient than ours. This shows a trade-off between the signature size and the online communication complexity.

Secondly, we compare ours with PW-1 and PW-2. The signature size of our scheme is reduced by more than 67% and 40% , respectively. The communication complexity of our scheme is also reduced by more than 57% and 41%. The public keys of them are two group elements. The public key size of PW-1 is 1028 bits and this scheme does not support key aggregation. The key size of PW-2 is 770 bits as same as ours. Thus, we also conclude that our scheme is more efficient than Pan-Wagner's schemes.

We implement our scheme on an ordinary machine and measure the running time of our implementation. 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 5.2. 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 an 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 conclude that our scheme has a realistic running time in practice.

Our Techniques for Constructing Proposed Scheme. To achieve a scheme with a small reduction loss, we start with the tightly secure Katz-Wang signature scheme [GJKW07] based on the DDH problem. Towards constructing a two-round multi-signature scheme, since there are some three-round multi-signature schemes based on the Katz-Wang signature scheme, we attempt to reduce the number of rounds of the signing protocol. Then, we can use the technique of a two-round scheme mBCJ to achieve the two-round signing protocol. This scheme has a two-round signing protocol by applying a special commitment scheme based on the DL problem to a DL-based three-round scheme. Since this approach is modular, we can construct a two-round signature scheme by constructing a suitable special commitment scheme for the DDH problem.

Unfortunately, the two-round scheme constructed by applying the modular way of mBCJ, e.g., PW-2, leads to the inefficient signature size. The cause of this is the inefficiency of a DDH-based special commitment scheme. Such a commitment scheme needs to have equivocability and binding prop-

Table 1.1: Signature size and communication complexity for two-round multi-signature schemes without using the AGM.

Scheme	Signature Size (bit)	Communication Comp. (bit)
MuSig-DN	1481	-
MuSig2-1	1501	3754 (750)
HBMS	2959	2959
TZ	2227	4456 (1484)
mBCJ-PPK	2297	2872
PW-1	3590+N	3591
PW-2	1920	2691
HBMSDDH-1	1152	1538

* Columns 2 and 3 show the signature size and the sum of the online and offline communication complexity, respectively. For MuSig2-1 and TZ, the values in () in the third column are the online communication complexity. For PW-1, PW-2, and HBMSDDH-1, we show the sizes of the multi-signature and elements sent in the signing protocol per a signer under the NIST standardized ECs for 128-bit security. For other schemes, we show the sizes of them under groups of the smallest order that guarantees 128-bit security. N indicates the number of signers. For communication complexity of MuSig-DN, we write "-" because it includes a proof generated by another cryptographic tool whose size considering concrete security is explicitly unknown.

erty, which are required to prove the security of the resulting two-round scheme. To ensure these properties, the commitment scheme requires a large commitment key and decommitment. This brings about a large size of the signature.

We then focus on the technique of a two-round scheme HBMS to resolve inefficiency. HBMS is a similar scheme to mBCJ but improves the signature size. Nevertheless, the cause of the improvement is not explained. However, upon our careful observation, the improvement is achieved by removing the binding property from the commitment scheme and simplifying it. In the security proof, the reduction embeds a special structure in commitment keys so that it can solve the DL problem from forgeries without binding property. Unfortunately, we cannot simply apply this technique to our case since the technique of HBMS is strongly dependent on the structure of the DL-based Schnorr signature scheme.

To overcome this challenge, we construct our scheme HBMSDDH-1 by tuning the approach observed in HBMS to the scheme based on the DDH problem. In short, we construct a special commitment scheme in which we can embed a special structure to prove the security of the DDH-based scheme instead of ensuring the binding property in the commitment keys. Then, we achieve a smaller commitment key and decommitment than those of the scheme based on mBCJ, e.g., PW-2. Specifically, while those of PW-2 consist of nine and three group elements, respectively, those of our scheme consist of two group elements and only one group element,respectively. Consequently, the signature of our scheme consists of three scalars, which is equivalent to HBMS. We prove that HBMSDDH-1 satisfies the slightly weak unforgeability under the DDH assumption. In this weak unforgeability, the forgery on an already signed message does not count as forgery, as opposed to the standard unforgeability, in which the forgery on an already signed message and a set of public keys does not count as forgery. We also propose a variant scheme HBMSDDH-2. This variant scheme achieves the standard unforgeability without compromising the efficiency and reliability of the security of the original scheme HBMSDDH-1.

1.5 Multi-Signatures Based on Other Computational Problem

RSA-Based Multi-Signatures. Although there are some RSA-based multi-signature schemes, all of them have special limitations. Most of them achieve only sequential signing protocol [Oka88, HK89, Oka93, PPKW97, DMO00, MM00]. While Desmedt and Frankel [DF92] proposed a non-interactive signing protocol, it requires a trusted third party who distributes all signers' secret keys. Recently, Tessaro and Zhu [TZ23] proposed not only a DL-based scheme but also an RSA-based scheme, in which the public parameters must be generated honestly.

Pairing-Based Multi-Signatures. Pairing provides a suitable algebraic structure for aggregating signatures. Indeed, there is a non-interactive aggregate signature scheme based on pairing [BGLS03], which aggregates multiple signatures on different messages into one signature. Multi-signatures are a special case of aggregate signatures. On the other hand, the security is less reliable compared to the DL-based schemes due to cryptanalytic efforts, e.g., [KB16, Gui20], and libraries for the implementation are not sufficiently supported.

The original pairing-based aggregate signature scheme [BGLS03] cannot be used as the multi-signatures since the security requires a condition where all messages are distinct. The variant scheme removes such the restriction and is compatible with multi-signatures [BNN07]. Boldyreva [Bol03] proposed a pairing-based multi-signature scheme that is secure in the knowledge of secret key (KOSK) model, and Ristenpart and Yilek [RY07] proved the security of the scheme in the proof-of-possession (PoP) model by applying a simple PoP protocol. Le et al. [LBG09] proposed a pairing-based three-round scheme that is secure in the PPK model. Lu et al. [LOS$^+$06] proposed a scheme without using the random oracle model. Boneh et al. [BDN18] proposed a scheme supporting key aggregation and achieving security in the PPK model and an accountable-subgroup multi-signature scheme. Drijvers et al. [DGNW20] constructed a forward-secure multi-signature scheme. While Kojima et al. [KSH23] constructs the two-round pairing-based multi-signature scheme supporting key aggregation, the security of this scheme is proven in the PoP model.

Lattice-Based Multi-Signatures. There are some three-round lattice-based multi-signatures [ES16, MJ19, FH20] following the Fiat-Shamir with abort paradigm [Lyu09, Lyu12]. Damgard et al. [DOTT21] constructed a two-round scheme by using an equivocal commitment scheme, like mBCJ. Boschini et al. [BTT22] proposed a two-round scheme that is similar to MuSig2. Chen [Che23b] proposed a more efficient two-round scheme that has an analogous structure to HBMS.

1.6 Organization

The remainder of this thesis is organized as follows: In Chapter 2, we explain the notations and recall a lemma, definitions of the computational problems and assumptions based on the discrete logarithm, and the syntaxes and security definitions of the multi-signatures. In Chapter 3, we first explain the basic DL-based signature scheme, e.g., the Schnorr signature scheme, and the key setup model, and then we review some DL-based multi-signature schemes. In Chapter 4, we propose our new two-round multi-signature schemes and show their security. In Chapter 5, we compare our proposed schemes with the related two-round schemes in concrete security, and also we evaluate the computation time and communication time. In Chapter 6, we discuss the result of this thesis. In Chapter 7, we describe the conclusion of this thesis.

Chapter 2

Preliminaries

In this chapter, we prepare the notations and recall a lemma, the definitions of the computation problems and assumptions, and syntaxes and security definitions of the multi-signatures.

Road Maps. In Section 2.1, we prepare the general notations. In Section 2.2, we recall some computational problems and assumptions based on the discrete logarithm. In Section 2.3, we recall the general forking lemma [BN06]. In Section 2.4, we show the syntaxes of multi-signature schemes and the definitions of the correctness and the unforgeability.

2.1 General Notations

We denote the security parameter by λ. Unless noted otherwise, any algorithm is probabilistic. For an algorithm $\mathcal{A}$, we write $b \stackrel{\$}{\leftarrow} \mathcal{A}(\alpha_1, \ldots)$ to mean that $\mathcal{A}$ takes as inputs $\alpha_1, \ldots$ and a uniformly chosen random tape and outputs b. For a list L, we write the i-th element in L as $L[i]$ and the size of L as $|L|$. For any tuple $\mathbf{a}$, the number of elements in $\mathbf{a}$ is denoted by $|\mathbf{a}|$. For any value a, we write $a \leftarrow b$ means the assignment of a into b.

For a prime integer p, we denote the ring of integers modulo p by $\mathbb{Z}_p$. Let $\mathbb{G}$ be an additive cyclic group of order p and let G be a generator of $\mathbb{G}$. We denote the identity element of $\mathbb{G}$ by O. Let GrGen be a probabilistic polynomial-time algorithm that takes as input a security parameter 1^λ and outputs a group description $(\mathbb{G}, p, G)$ consisting of a group $\mathbb{G}$ of order p, where p is a prime and $\log p = \Omega(\lambda)$, and G is a generator of $\mathbb{G}$.

For $A, B, G, H \in \mathbb{G}$ and $x \in \mathbb{Z}_p$, we write $(A, B)^\top \leftarrow x(G, H)^\top$ to mean that A and B are computed by xG and xH, respectively. Also, for $A, B, G, H, X, Y \in \mathbb{G}$, we write $(A, B)^\top \leftarrow (G, H)^\top + (X, Y)^\top$ to mean that A and B are computed by $G + X$ and $H + Y$, respectively.

2.2 Discrete Logarithm

2.2.1 Problems and Assumptions

Below, we recall the definitions of the discrete logarithm (DL) problem, the decisional Diffie-Hellman (DDH) problem, and the algebraic one-more discrete logarithm (AOMDL) problem.

Definition 1 (Discrete Logarithm Problem)**.** *The advantage of an adversary $\mathcal{A}$ for the discrete logarithm (DL) problem is defined by*

$$\mathsf{Adv}^{\mathrm{dl}}_{\mathcal{A}}(1^\lambda) = \Pr[\mathsf{Game}^{\mathrm{dl}}_{\mathcal{A}}(1^\lambda) = 1].$$

We say that an adversary $\mathcal{A}$ (t, ϵ)-solves the DL problem if it runs in time at most t and satisfies $\mathsf{Adv}^{\mathrm{dl}}_{\mathcal{A}}(1^\lambda) \geq \epsilon$. We also say that the DL assumption holds if $\mathsf{Adv}^{\mathrm{dl}}_{\mathcal{A}}(1^\lambda)$ is negligible for any PPT algorithm $\mathcal{A}$.

Definition 2 (Decisional Diffie-Hellman Problem)**.** *The advantage of an adversary $\mathcal{A}$ for the decisional Diffie-Hellman (DDH) problem is defined by*

$$\mathsf{Adv}^{\mathrm{ddh}}_{\mathcal{A}}(1^\lambda) = \Pr[\mathsf{Game}^{\mathrm{ddh}}_{\mathcal{A}}(1^\lambda) = 1].$$

We say that an adversary $\mathcal{A}$ (t, ϵ)-solves the DDH problem if it runs in time at most t and satisfies $\mathsf{Adv}^{\mathrm{ddh}}_{\mathcal{A}}(1^\lambda) \geq \epsilon$. We say that the DDH assumption holds if $\mathsf{Adv}^{\mathrm{ddh}}_{\mathcal{A}}(1^\lambda)$ is negligible for any PPT adversary $\mathcal{A}$. We also say that $\mathbb{G}$ is a (t, ϵ)-DDH group if there is no adversary $\mathcal{A}$ that (t, ϵ)-solves the DDH problem.

For $G, H, X, Y \in \mathbb{G}$, the tuple (G, H, X, Y) is called a DH-tuple (resp. non-DH tuple) if there exists (resp. does not exist) $x \in \mathbb{Z}_p$ such that $x(G, H)^\top = (X, Y)^\top$.

The AOMDL problem is a variant of the OMDL problem introduced by [NRS21]. The difference between them is queries to the DL oracle. An adversary against the OMDL problem is allowed to query any group elements in $\mathbb{G}$ to the DL oracle, which returns the discrete logarithm of the queried group element. Notice that the OMDL assumption is unfalsifiable since the oracle needs to solve the DL problem to answer queries. On the other hand, an adversary against the AOMDL problem is only allowed to query linear combinations of group elements given as a challenge instance. The AOMDL assumption is falsifiable since the challenger can know all discrete logarithms of challenge group elements.

Definition 3 (Algebraic One-More Discrete Logarithm Problem [NRS21]). *The advantage of an adversary $\mathcal{A}$ for the ℓ algebraic one-more discrete logarithm (ℓ-AOMDL) problem is defined by*

$$\mathsf{Adv}^{\ell\text{-aomdl}}_{\mathcal{A}}(1^\lambda) = \Pr[\mathsf{Game}^{\ell\text{-aomdl}}_{\mathcal{A}}(1^\lambda) = 1].$$

We say that an adversary $\mathcal{A}$ (t, ϵ)-solves the ℓ-AOMDL problem if it runs in time at most t and satisfies $\mathsf{Adv}^{\ell\text{-aomdl}}_{\mathcal{A}}(1^\lambda) \geq \epsilon$. We also say that the ℓ-AOMDL assumption holds if $\mathsf{Adv}^{\ell\text{-aomdl}}_{\mathcal{A}}(1^\lambda)$ is negligible for any PPT adversary $\mathcal{A}$.

$\mathsf{Game}^{\mathrm{dl}}_{\mathcal{A}}(1^\lambda)$:

$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$
$x \xleftarrow{\$} \mathbb{Z}_p, X \leftarrow xG$
$x' \leftarrow \mathcal{A}((\mathbb{G}, p, G), X)$
return $(x = x')$

$\mathsf{Game}^{\mathrm{ddh}}_{\mathcal{A}}(1^\lambda)$:

$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$
$b \xleftarrow{\$} \{0, 1\}$
$x, y, \xleftarrow{\$} \mathbb{Z}_p, z \leftarrow \mathbb{Z}_p \backslash \{xy\}$
$X \leftarrow xG, Y \leftarrow yG$
if $b = 0$,
  $Z \leftarrow xyG$
else
  $Z \leftarrow zG$
$b' \leftarrow \mathcal{A}((\mathbb{G}, p, G), X, Y, Z)$
return $(b = b')$

$\mathsf{Game}^{\ell\text{-aomdl}}_{\mathcal{A}}(1^\lambda)$:

$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$
$\mathsf{ctr} \leftarrow 0$
for $i \in [0, \ell]$ **do**
  $x_i \xleftarrow{\$} \mathbb{Z}_p, X_i \leftarrow xG$
$\mathbf{X} \leftarrow (X_0, X_1, \ldots, X_\ell)$
$(x'_i)_{i=0}^{\ell} \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{dl}}}((\mathbb{G}, p, G), \mathbf{X})$
return $((x_i)_{i=0}^{\ell} = (x'_i)_{i=0}^{\ell}) \wedge (\mathsf{ctr} < \ell)$

$\mathcal{O}_{\mathrm{dl}}((a_i)_{i=0}^{\ell})$:

$\mathsf{ctr} \leftarrow \mathsf{ctr} + 1$
$s \leftarrow \sum_{i=0}^{\ell} a_i x_i$
return s

Figure 2.1: The game of the DL problem, the DDH problem, and the AOMDL problem.

Now we recall the definition of the algebraic group model (AGM) [FKL18]. This is an idealized model for computations over a group $\mathbb{G}$. In short, this assumption states that an adversary can only produce a new group element by linearly combining group elements that it has seen so far.

Definition 4 (Algebraic Group Model [FKL18]). *An adversary $\mathcal{A}$ is algebraic if for every group element $X \in \mathbb{G}$ that it outputs, it is required to output a representation $\mathbf{a} = (a_0, a_1, \dots) \in \mathbb{Z}_p^{|\mathbf{a}|}$ such that $X = a_0 G + \sum_i a_i Y_i$ where $Y_1, Y_2, \dots \in \mathbb{G}$ are group elements that $\mathcal{A}$ has seen so far.*

2.2.2 Randomizing Algorithm of (non-)DH Tuple

Bellare et al. proposed a randomizing algorithm of a (non-)DH tuple in [BBM00]. Their algorithm on input a (non-)DH tuple outputs a re-randomized (non-)DH tuple. More concretely, the algorithm is given a tuple $(G, H, P, Q) \in \mathbb{G}^4$ as input and outputs a tuple $(G, H', P', Q') \in \mathbb{G}^4$. If (G, H, P, Q) is a DH tuple, (G, H', P', Q') satisfies that (H', P') is uniformly distributed over $\mathbb{G}^2$ and (G, H', P', Q') is a DH tuple. If (G, H, P, Q) is a non-DH tuple, (G, H', P', Q') satisfies that (H', P', Q') is uniformly distributed over $\mathbb{G}^3$.

In this thesis, we use the subtly modified algorithm RandDH. This algorithm on input a (non-)DH tuple outputs a re-randomized (non-)DH tuple of which the second element is also the same as the second one of a tuple given as input. Specifically, if (G, H, P, Q) is a DH tuple, (G, H, P', Q') satisfies that P' is uniformly distributed over $\mathbb{G}$ and (G, H, P', Q') is a DH tuple. If (G, H, P, Q) is a non-DH tuple, (G, H, P', Q') satisfies that (P', Q') is uniformly distributed over $\mathbb{G}^2$. The description of this algorithm is shown in Fig. 2.2.

$\mathsf{RandDH}(G, H, P, Q)$:
1 : $s, t \xleftarrow{\$} \mathbb{Z}_p$
2 : $P' \leftarrow sG + tP, Q' \leftarrow sH + tQ$
3 : **return** (P', Q')

Figure 2.2: The description of the randomizing algorithm of (non-)DH tuple.

2.3 General Forking Lemma

Here, we review the General Forking Lemma [BN06].

Lemma 1 (General Forking Lemma [BN06]). *Let $Q \geq 1$ be an integer, and $\mathcal{H}$ be a set of size $|\mathcal{H}| \geq 2$, where $|\mathcal{H}|$ is the size of $\mathcal{H}$. Let* IG *be a randomized* *algorithm that is called the input generator and* $\mathcal{A}$ *be a randomized algorithm that, on input* $(\mathsf{par}, h_1, \ldots, h_Q)$ *where* par *is an input of the forking lemma generated by* IG *and* $h_i \in \mathcal{H}$ *for* $^\forall i \in [1, Q]$*, returns* $(I, \sigma) \in [0, Q] \times \{0,1\}^*$*. The accepting probability of* $\mathcal{A}$*, denoted* acc*, is defined as the probability that* $J \geq 1$ *in the experiment* $\mathsf{par} \xleftarrow{\$} \mathsf{IG}, \rho \xleftarrow{\$} R, h_1, \ldots, h_Q \xleftarrow{\$} \mathcal{H}, (J, \sigma) \leftarrow \mathcal{A}(\mathsf{par}, h_1, \ldots, h_Q; \rho)$ *where* R *is the set of random tapes. The forking algorithm* $\mathsf{Fork}_{\mathcal{A}}(\mathsf{par})$ *associated to* $\mathcal{A}$ *is the randomized algorithm that takes input* par *proceeds as in Fig. 2.3. Let*

$$\mathsf{frk} = \Pr[b = 1 : \mathsf{par} \xleftarrow{\$} \mathsf{IG}, (b, \sigma, \sigma') \xleftarrow{\$} \mathsf{Fork}_{\mathcal{A}}(\mathsf{par})].$$

Then,

$$\mathsf{frk} \geq \mathsf{acc} \cdot \left(\frac{\mathsf{acc}}{Q} - \frac{1}{|\mathcal{H}|} \right).$$

Algorithm $\mathsf{Fork}_{\mathcal{A}}(\mathsf{par})$

$\rho \xleftarrow{\$} R$
$h_1, \ldots, h_Q \xleftarrow{\$} \mathcal{H}$
$(J, \sigma) \leftarrow \mathcal{A}(\mathsf{par}, h_1, \ldots, h_Q; \rho)$
if $J = 0$
  return $(0, \perp, \perp)$
$h'_J, \ldots, h'_Q \xleftarrow{\$} \mathcal{H}$
$(J', \sigma') \leftarrow \mathcal{A}(\mathsf{par}, h_1, \ldots, h_{J-1}, h'_J, \ldots, h'_Q; \rho)$
if $(J = J' \wedge h_J \neq h_{J'})$
  return $(1, \sigma, \sigma')$
else
  return $(0, \perp, \perp)$

Figure 2.3: Description of the forking algorithm $\mathsf{Fork}_{\mathcal{A}}$.

2.4 Definition of Multi-Signatures

In this section, we show the syntaxes and security definitions for multi-signatures. Specifically, we show the syntaxes of a two-round multi-signature scheme, a one-round multi-signature scheme with pre-processing, and a three-round multi-signature scheme. The difference between them is the signing protocol. Specifically, a two-round multi-signature scheme has two signing algorithms for the first round and the second round. On the other hand, a one-round multi-signature scheme with pre-processing has two algorithms for the pre-processing and the signing. Moreover, a three-round multi-signature scheme has three signing algorithms for the first, second, and third rounds. For correctness and unforgeability, we show the definition of them for each type of multi-signature scheme.

2.4.1 Syntax

We first show the syntaxes for each type of multi-signature scheme, i.e., a two-round multi-signature scheme, a one-round multi-signature scheme with pre-processing, and a three-round multi-signature scheme. Since they have the same syntax except for signing algorithms, we describe the signing algorithm for each type of multi-signature scheme.

A two-round multi-signature scheme, a one-round multi-signature scheme with pre-processing, and a three-round multi-signature scheme consist of the following algorithms. Let N be the number of signers, namely $N = |\mathsf{vkList}|$.

$\mathsf{Setup}(1^\lambda) \to \mathsf{par}$**.** The setup algorithm takes as input the security parameter 1^λ and outputs a public parameter par.

$\mathsf{KeyGen}(\mathsf{par}) \to (\mathsf{pk}, \mathsf{sk})$**.** The key generation algorithm takes as input a public parameter par and outputs a public key pk and a secret key sk.

Two-Round: $\mathsf{Sign}^{(2)} = (\mathsf{Sign}_1^{(2)}, \mathsf{Sign}_2^{(2)})$**.** The signing protocol Sign of a two-round multi-signature scheme consists of the following two algorithms $\mathsf{Sign}_1^{(2)}$ and $\mathsf{Sign}_2^{(2)}$.

$\mathsf{Sign}_1^{(2)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i) \to (\mathsf{pm}_i, \mathsf{st}_i)$**.** The signing algorithm for the first round takes as input a public parameter par, a public key list $\mathsf{vkList} = (\mathsf{pk}_j)_{j \in [N]}$, a message M to be signed, an index i of the signer, and a secret key sk_i of signer i, and outputs a protocol message pm_i and a state st_i.

$\mathsf{Sign}_2^{(2)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_j)_{j \in [N] \setminus \{i\}}) \to \mathsf{psig}_i$**.** The signing algorithm for the second round takes as input a public parameter par, a public key list vkList, a message M to be signed, an index i of the signer, a secret key sk_i and a state st_i of signer i, and a tuple of protocol messages $(\mathsf{pm}_j)_{j \in [N] \setminus \{i\}}$ and outputs a partial signature psig_i.

One-Round with Pre-Processing: $\mathsf{Sign}^{(1\text{-}1)} = (\mathsf{Sign}_0^{(1\text{-}1)}, \mathsf{Sign}_1^{(1\text{-}1)})$**.** The signing protocol Sign of a one-round multi-signature scheme with pre-processing consists of the following two algorithms $\mathsf{Sign}_0^{(1\text{-}1)}$ and $\mathsf{Sign}_1^{(1\text{-}1)}$.

$\mathsf{Sign}_0^{(1\text{-}1)}(\mathsf{par}, i, \mathsf{sk}_i) \to (\mathsf{pm}_i, \mathsf{st}_i)$**.** The signing algorithm for pre-processing takes as input a public parameter par, an index i of the signer, and a secret key sk_i of signer i, and outputs a protocol message pm_i and a state st_i.

$\mathsf{Sign}_1^{(1\text{-}1)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_j)_{j\in[N]\setminus\{i\}}) \to \mathsf{psig}_i$**.** The signing algorithm for signing takes as input a public parameter par, a public key list vkList, a message M to be signed, an index i of the signer, a secret key sk_i and a state st_i of signer i, and a tuple of protocol messages $(\mathsf{pm}_j)_{j\in[N]\setminus\{i\}}$ and outputs a partial signature psig_i.

Three-Round: $\mathsf{Sign}^{(3)} = (\mathsf{Sign}_1^{(3)}, \mathsf{Sign}_2^{(3)}, \mathsf{Sign}_3^{(2)})$**.** The signing protocol Sign of a three-round multi-signature scheme consists of the following three algorithms $\mathsf{Sign}_1^{(3)}$, $\mathsf{Sign}_2^{(3)}$, and $\mathsf{Sign}_3^{(2)}$.

$\mathsf{Sign}_1^{(3)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i) \to (\mathsf{pm}_{1,i}, \mathsf{st}_i)$**.** The signing algorithm for the first round takes as input a public parameter par, a public key list vkList, a message M to be signed, an index i of the signer, and a secret key sk_i of signer i, and outputs a protocol message $\mathsf{pm}_{1,i}$ and a state st_i.

$\mathsf{Sign}_2^{(3)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_{1,j})_{j\in[N]\setminus\{i\}}) \to (\mathsf{pm}_{2,i}, \mathsf{st}_i)$**.** The signing algorithm for the second round takes as input a public parameter par, a public key list vkList, a message M to be signed, an index i of the signer, a secret key sk_i and a state st_i of signer i, and a tuple of protocol messages $(\mathsf{pm}_{1,j})_{j\in[N]\setminus\{i\}}$ and outputs a protocol message $\mathsf{pm}_{2,i}$ and a state st_i.

$\mathsf{Sign}_3^{(3)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_{2,j})_{j\in[N]\setminus\{i\}}) \to \mathsf{psig}_i$**.** The signing algorithm for the third round takes as input a public parameter par, a public key list vkList, a message M to be signed, an index i of the signer, a secret key sk_i and a state st_i of signer i, and a tuple of protocol messages $(\mathsf{pm}_{2,j})_{j\in[N]\setminus\{i\}}$ and outputs a partial signature psig_i.

$\mathsf{Agg}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, (\mathsf{pm}_i, \mathsf{psig}_i)_{i\in[N]}) \to \widetilde{\mathsf{sig}}$**.** The aggregation algorithm takes as input a public parameter par, a public key list vkList, a message M to be signed, and all signers' protocol messages and partial signatures $(\mathsf{pm}_i, \mathsf{psig}_i)_{i\in[N]}$ and deterministically outputs a multi-signature. Note that pm_i for a three-round multi-signature scheme includes both of the protocol messages $\mathsf{pm}_{1,i}$ and $\mathsf{pm}_{2,i}$.

$\mathsf{Verify}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, \widetilde{\mathsf{sig}}) \rightarrow \{0, 1\}$. The verification algorithm takes as inputs a public parameter par, a public key list vkList, a message M to be signed, and a multi-signature $\widetilde{\mathsf{sig}}$ and deterministically outputs 1 (Accept) or 0 (Reject).

2.4.2 Correctness

Now we define the correctness for each type of multi-signature schemes.

Definition 5 (Correctness for Two-Round Multi-Signature Scheme). *We say that a two-round multi-signature scheme $\mathsf{MS}^{(2)}$ satisfies correctness if, for all $\lambda \in \mathbb{N}$, $N \in \mathsf{poly}(\lambda)$, positive integer $n \leq N$, and message M, the following holds:*

$$\Pr\left[\mathsf{Game}^{\mathsf{ms}^2\text{-}\mathsf{cor}}_{\mathsf{MS}^{(2)}}(1^\lambda, n, \mathsf{M}) = 1\right] = 1,$$

where $\mathsf{Game}^{\mathsf{ms}^2\text{-}\mathsf{cor}}_{\mathsf{MS}^{(2)}}$ is shown in Fig. 2.4.

Definition 6 (Correctness for One-Round Multi-Signature Scheme with Pre-Processing). *We say that a one-round multi-signature scheme with pre-processing $\mathsf{MS}^{(1\text{-}1)}$ satisfies correctness if, for all $\lambda \in \mathbb{N}$, $N \in \mathsf{poly}(\lambda)$, positive integer $n \leq N$, and message M, the following holds:*

$$\Pr\left[\mathsf{Game}^{\mathsf{ms}^{1\text{-}1}\text{-}\mathsf{cor}}_{\mathsf{MS}^{(1\text{-}1)}}(1^\lambda, n, \mathsf{M}) = 1\right] = 1,$$

where $\mathsf{Game}^{\mathsf{ms}^{1\text{-}1}\text{-}\mathsf{cor}}_{\mathsf{MS}^{(1\text{-}1)}}$ is shown in Fig. 2.5.

Definition 7 (Correctness for Three-Round Multi-Signature Scheme). *We say that a three-round multi-signature scheme $\mathsf{MS}^{(3)}$ satisfies correctness if, for all $\lambda \in \mathbb{N}$, $N \in \mathsf{poly}(\lambda)$, positive integer $n \leq N$, and message M, the following holds:*

$$\Pr\left[\mathsf{Game}^{\mathsf{ms}^3\text{-}\mathsf{cor}}_{\mathsf{MS}^{(3)}}(1^\lambda, n, \mathsf{M}) = 1\right] = 1,$$

where $\mathsf{Game}^{\mathsf{ms}^3\text{-}\mathsf{cor}}_{\mathsf{MS}^{(3)}}$ is shown in Fig. 2.6.

$\mathsf{Game}^{\mathsf{ms^2\text{-}cor}}_{\mathsf{MS}}(1^\lambda, n, \mathsf{M})$:

$\mathsf{par} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$

for $i \in [n]$ **do**

  $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{KeyGen}(\mathsf{par})$

for $i \in [n]$ **do**

  $(\mathsf{pm}_i, \mathsf{st}_i) \xleftarrow{\$} \mathsf{Sign}^{(2)}_1(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i)$

for $i \in [n]$ **do**

  $\mathsf{psig}_i \xleftarrow{\$} \mathsf{Sign}^{(2)}_2(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_j)_{j \in [n] \setminus \{j\}})$

$\widetilde{\mathsf{sig}} \leftarrow \mathsf{Agg}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, (\mathsf{pm}_i, \mathsf{psig}_i)_{i \in [n]})$

return $\mathsf{Verify}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, \widetilde{\mathsf{sig}})$

Figure 2.4: Correctness game for a two-round multi-signature scheme.

2.4.3 Unforgeability

Here we show the definition of the existential unforgeability under a chosen message attack for a multi-signature scheme. The notion of unforgeability requires that any adversary be infeasible to forge multi-signature involving at least one honest signer. An adversary is allowed to corrupt all signers except for one honest signer. It can query a message and a public-key list including at least one challenge key to the signing oracle and executes the signing protocol with the oracle by (maliciously) behaving as all cosigners. The goal of an adversary is to output a non-trivial valid multi-signature as a forgery. Note that "non-trivial" means that it is required to be forgery on a pair of a message and a public-key list which does not appear in signing queries. Moreover, it can maliciously choose all cosigners' public keys.

We show the unforgeability game for each type of multi-signature scheme. Note that all definitions are in the random oracle. We omit the description of the random oracle. Without loss of generality, we suppose that the honest signer corresponding to the challenge key has the signer index 1. The unforgeability games for each type are depicted in Figs. 2.7 to 2.9.

Definition 8 (Unforgeability for Two-Round Multi-Signature Scheme)**.** *For a two-round multi-signature signature scheme* $\mathsf{MS}^{(2)}$*, the advantage of an adversary* $\mathcal{A}$ *against the unforgeability of* $\mathsf{MS}^{(2)}$ *in the random oracle model is defined as*

$$\mathsf{Adv}^{\mathsf{ms^2\text{-}uf}}_{\mathsf{MS}^{(2)},\mathcal{A}}(1^\lambda, N) = \Pr\left[\mathsf{Game}^{\mathsf{ms^2\text{-}uf}}_{\mathsf{MS}^{(2)},\mathcal{A}}(1^\lambda, N) = 1\right],$$

$\mathsf{Game}^{\mathsf{ms}^{1\text{-}1}\text{-}\mathsf{cor}}_{\mathsf{MS}}(1^\lambda, n, \mathsf{M})$:

$\mathsf{par} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
for $i \in [n]$ **do**
  $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{KeyGen}(\mathsf{par})$
for $i \in [n]$ **do**
  $(\mathsf{pm}_i, \mathsf{st}_i) \xleftarrow{\$} \mathsf{Sign}_0^{(1\text{-}1)}(\mathsf{par}, i, \mathsf{sk}_i)$
for $i \in [n]$ **do**
  $\mathsf{psig}_i \xleftarrow{\$} \mathsf{Sign}_1^{(1\text{-}1)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_j)_{j \in [n] \setminus \{j\}})$
$\widetilde{\mathsf{sig}} \leftarrow \mathsf{Agg}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, (\mathsf{pm}_i, \mathsf{psig}_i)_{i \in [n]})$
return $\mathsf{Verify}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, \widetilde{\mathsf{sig}})$

Figure 2.5: Correctness game for a one-round multi-signature scheme with pre-processing.

$\mathsf{Game}^{\mathsf{ms}^{3}\text{-}\mathsf{cor}}_{\mathsf{MS}}(1^\lambda, n, \mathsf{M})$:

$\mathsf{par} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
for $i \in [n]$ **do**
  $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{KeyGen}(\mathsf{par})$
for $i \in [n]$ **do**
  $(\mathsf{pm}_{1,i}, \mathsf{st}_i) \xleftarrow{\$} \mathsf{Sign}_1^{(3)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i)$
for $i \in [n]$ **do**
  $(\mathsf{pm}_{2,i}, \mathsf{st}_i) \leftarrow \mathsf{Sign}_2^{(3)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_{1,j})_{j \in [n] \setminus \{j\}})$
for $i \in [n]$ **do**
  $\mathsf{psig}_i \xleftarrow{\$} \mathsf{Sign}_3^{(3)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_{2,j})_{j \in [n] \setminus \{j\}})$
  $\mathsf{pm}_i := (\mathsf{pm}_{1,i}, \mathsf{pm}_{2,i})$
$\widetilde{\mathsf{sig}} \leftarrow \mathsf{Agg}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, (\mathsf{pm}_i, \mathsf{psig}_i)_{i \in [n]})$
return $\mathsf{Verify}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, \widetilde{\mathsf{sig}})$

Figure 2.6: Correctness game for a three-round multi-signature scheme.

where $\mathsf{Game}^{\mathsf{ms}^2\text{-}\mathsf{uf}}_{\mathsf{MS}^{(2)},\mathcal{A}}(1^\lambda, N)$ *is described in Fig. 2.7. We say that* $\mathcal{A}$ *is* $(t, Q_S, Q_H, N, \epsilon)$*-breaks the 2-MS-UF of* $\mathsf{MS}^{(2)}$ *if* $\mathcal{A}$ *runs in at most* t *time, makes at most* Q_S *signing queries and* Q_H *random oracle queries, and* $\mathsf{Adv}^{\mathsf{ms}^2\text{-}\mathsf{uf}}_{\mathsf{MS}^{(2)},\mathcal{A}}(1^\lambda, N) \geq \epsilon$

for all $\lambda \in \mathbb{N}$*. We also say* $\mathsf{MS}^{(2)}$ *is* $(t, Q_S, Q_H, N, \epsilon)$*-2-MS-UF if there is no* $\mathcal{A}$ *that* $(t, Q_S, Q_H, N, \epsilon)$*-breaks the 2-MS-UF of* $\mathsf{MS}^{(2)}$*.*

Definition 9 (Unforgeability for One-Round Multi-Signature Scheme with Pre-Processing)**.** *For a one-round multi-signature signature scheme* $\mathsf{MS}^{(1\text{-}1)}$*, the advantage of an adversary* $\mathcal{A}$ *against the unforgeability of* $\mathsf{MS}^{(1\text{-}1)}$ *in the random oracle model is defined as*

$$\mathsf{Adv}^{\mathsf{ms}^{1\text{-}1}\text{-}\mathsf{uf}}_{\mathsf{MS}^{(1\text{-}1)},\mathcal{A}}(1^\lambda, N) = \Pr\left[\mathsf{Game}^{\mathsf{ms}^{1\text{-}1}\text{-}\mathsf{uf}}_{\mathsf{MS}^{(1\text{-}1)},\mathcal{A}}(1^\lambda, N) = 1\right],$$

where $\mathsf{Game}^{\mathsf{ms}^{1\text{-}1}\text{-}\mathsf{uf}}_{\mathsf{MS}^{(3)},\mathcal{A}}(1^\lambda, N)$ *is described in Fig. 2.8. We say that* $\mathcal{A}$ *is* $(t, Q_S, Q_H, N, \epsilon)$*-breaks the (1-1)-MS-UF of* $\mathsf{MS}^{(1\text{-}1)}$ *if* $\mathcal{A}$ *runs in at most* t *time, makes at most* Q_S *signing queries and* Q_H *random oracle queries, and* $\mathsf{Adv}^{\mathsf{ms}^{1\text{-}1}\text{-}\mathsf{uf}}_{\mathsf{MS}^{(1\text{-}1)},\mathcal{A}}(1^\lambda, N) \geq \epsilon$ *for all* $\lambda \in \mathbb{N}$*. We also say* $\mathsf{MS}^{(1\text{-}1)}$ *is* $(t, Q_S, Q_H, N, \epsilon)$*-(1-1)-MS-UF if there is no* $\mathcal{A}$ *that* $(t, Q_S, Q_H, N, \epsilon)$*-breaks the (1-1)-MS-UF of* $\mathsf{MS}^{(1\text{-}1)}$*.*

Definition 10 (Unforgeability for Three-Round Multi-Signature Scheme)**.** *For a three-round multi-signature signature scheme* $\mathsf{MS}^{(3)}$*, the advantage of an adversary* $\mathcal{A}$ *against the unforgeability of* $\mathsf{MS}^{(3)}$ *in the random oracle model is defined as*

$$\mathsf{Adv}^{\mathsf{ms}^{3}\text{-}\mathsf{uf}}_{\mathsf{MS}^{(3)},\mathcal{A}}(1^\lambda, N) = \Pr\left[\mathsf{Game}^{\mathsf{ms}^{3}\text{-}\mathsf{uf}}_{\mathsf{MS}^{(3)},\mathcal{A}}(1^\lambda, N) = 1\right],$$

where $\mathsf{Game}^{\mathsf{ms}^{3}\text{-}\mathsf{uf}}_{\mathsf{MS}^{(3)},\mathcal{A}}(1^\lambda, N)$ *is described in Fig. 2.9. We say that* $\mathcal{A}$ *is* $(t, Q_S, Q_H, N, \epsilon)$*-breaks the 3-MS-UF of* $\mathsf{MS}^{(3)}$ *if* $\mathcal{A}$ *runs in at most* t *time, makes at most* Q_S *signing queries and* Q_H *random oracle queries, and* $\mathsf{Adv}^{\mathsf{ms}^{3}\text{-}\mathsf{uf}}_{\mathsf{MS}^{(3)},\mathcal{A}}(1^\lambda, N) \geq \epsilon$ *for all* $\lambda \in \mathbb{N}$*. We also say* $\mathsf{MS}^{(3)}$ *is* $(t, Q_S, Q_H, N, \epsilon)$*-3-MS-UF if there is no* $\mathcal{A}$ *that* $(t, Q_S, Q_H, N, \epsilon)$*-breaks the 3-MS-UF of* $\mathsf{MS}^{(3)}$*.*

2.4.4 Slightly Weak and Strong Unforgeability

We use slightly different definitions of unforgeability from the conventional one when we prove the security of our schemes, which will be proposed later in Chapter 4. Specifically, we use the slightly strong definition and the slightly weak definition. The strong one slightly relaxes the requirement of the forgery and allows an adversary to obtain multiple partial signatures generated from the challenge key in one signing query. In the weak one, the goal of an adversary is slightly raised. Below, we first explain the two differences between the conventional definition and the strong one.

First, we explain the subtle difference in the winning conditions. The challenger in our unforgeability game counts $(\mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*)$ as a successful forgery even if a signing protocol for $(\mathsf{vkList}^*, \mathsf{M}^*)$ is opened but not completed. In other words, the forgery is valid even if $(\mathsf{vkList}^*, \mathsf{M}^*)$ is queried to the signing oracle as long as any signature on $(\mathsf{vkList}^*, \mathsf{M}^*)$ has never been received. In the conventional game, the outputs of an adversary do not count as a forgery. This modification captures adversaries who exploit the interruption of the signing protocol. To see the difference, let us consider the following example. An adversary sends a pair $(\mathsf{vkList}^*, \mathsf{M}^*)$ to the signing oracle as a signing query and receives a response of the first round from the oracle. Then, it outputs a forgery on $(\mathsf{vkList}^*, \mathsf{M}^*)$ without completing the signing protocol. If the forgery on $(\mathsf{vkList}^*, \mathsf{M}^*)$ is valid, it wins.

Next, we describe the difference in the signing oracle. We consider the case where an adversary makes a signing query whose public-key list vkList includes multiple challenge public keys. In the conventional game, the signing oracle responds by behaving as one honest signer. In our game, it responds by behaving as all honest signers who have the same public key. This modification captures the situation where the signing protocol is executed by a set of signers including some honest signers with the same public key.

Finally, we explain the difference between the strong one and the weaker one. The difference is that, in the game of the weak one, an adversary's output $(\mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*)$ does not count as a successful forgery if an adversary has received a signature on the message M^* from the signing oracle. In the game of the strong one, $(\mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*)$ counts as a forgery even if an adversary has ever received a signature on M^* as long as the forger has received a signature on $(\mathsf{vkList}^*, \mathsf{M}^*)$. Our first scheme will be proven secure under the slightly weaker unforgeability game. After that, we will modify our first scheme and show that the modified scheme achieves slightly strong unforgeability.

We show the slightly modified definitions of unforgeability for the two-round multi-signature scheme.

Definition 11 (Slightly Modified Unforgeability for Two-round Multi-Signature Scheme)**.** *For a two-round multi-signature signature scheme $\mathsf{MS}^{(2)}$, the advantages of an adversary $\mathcal{A}$ against the slightly strong and weak unforgeability of $\mathsf{MS}^{(2)}$ in the random oracle model are respectively defined as*

$$\mathsf{Adv}^{\mathsf{ms^2\text{-}uf1}}_{\mathsf{MS}^{(2)},\mathcal{A}}(1^\lambda, N) = \Pr[\mathsf{Game}^{\mathsf{ms^2\text{-}uf1}}_{\mathsf{MS}^{(2)},\mathcal{A}}(1^\lambda, N) = 1],$$
$$\text{and } \mathsf{Adv}^{\mathsf{ms^2\text{-}uf2}}_{\mathsf{MS}^{(2)},\mathcal{A}}(1^\lambda, N) = \Pr[\mathsf{Game}^{\mathsf{ms^2\text{-}uf2}}_{\mathsf{MS}^{(2)},\mathcal{A}}(1^\lambda, N) = 1],$$

where $\mathsf{Game}^{\mathsf{ms^2\text{-}uf1}}_{\mathsf{MS}^{(2)},\mathcal{A}}(1^\lambda, N)$ and $\mathsf{Game}^{\mathsf{ms^2\text{-}uf2}}_{\mathsf{MS}^{(2)},\mathcal{A}}(1^\lambda, N)$ are described in Fig. 2.10.

We say that adversary $\mathcal{A}$ *is* $(t, Q_S, Q_H, N, \epsilon)$*-breaks the 2-MS-UF-1 (resp.* $(t, Q_S, Q_H, N, \epsilon)$*-breaks the 2-MS-UF-2) of* $\mathsf{MS}^{(2)}$ *if* $\mathcal{A}$ *runs in at most* t *time, makes at most* Q_S *signing queries and* Q_H *random oracle queries, and* $\mathsf{Adv}^{\mathsf{ms^2\text{-}uf1}}_{\mathsf{MS}^{(2)},\mathcal{A}}(1^\lambda, N) \geq \epsilon$ *(resp.* $\mathsf{Adv}^{\mathsf{ms^2\text{-}uf2}}_{\mathsf{MS}^{(2)},\mathcal{A}}(1^\lambda, N) \geq \epsilon$*) for all* $\lambda \in \mathbb{N}$*. We also say* $\mathsf{MS}^{(2)}$ *is* $(t, Q_S, Q_H, N, \epsilon)$*-2-MS-UF-1 (resp.* $(t, Q_S, Q_H, N, \epsilon)$*-2-MS-UF-2) if there is no* $\mathcal{A}$ *that* $(t, Q_S, Q_H, N, \epsilon)$*-breaks the 2-MS-UF-1 (resp.* $(t, Q_S, Q_H, N, \epsilon)$*-breaks the 2-MS-UF-2) of* $\mathsf{MS}^{(2)}$*.*

$$\mathsf{Game}^{\mathsf{ms^2\text{-}uf}}_{\mathsf{MS}^{(2)},\mathcal{A}}(1^\lambda, N)$$

$\mathsf{Q_M} \leftarrow \emptyset, \mathsf{Q_{st}}[\cdot] \leftarrow \bot$

$\mathsf{par} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$

$(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}(\mathsf{par})$

$(\mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}_1^{(2)}}, \mathcal{O}_{\mathsf{Sign}_2^{(2)}}, \mathsf{H}}(\mathsf{par}, \mathsf{pk})$

$\textbf{req } [\![\mathsf{pk} \in \mathsf{vkList}^*]\!] \wedge [\![|\mathsf{vkList}^*| \leq N]\!] \wedge [\![(\mathsf{vkList}^*, \mathsf{M}^*) \notin \mathsf{Q_M}]\!]$

$\textbf{return } \mathsf{Verify}(\mathsf{par}, \mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*)$

$$\mathcal{O}_{\mathsf{Sign}_1^{(2)}}(\mathsf{sid}, \mathsf{vkList}, \mathsf{M})$$

$\textbf{req } [\![\mathsf{pk} \in \mathsf{vkList}]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 1] = \bot]\!] \wedge [\![|\mathsf{vkList}| \leq N]\!]$

$\mathsf{Q_M} \leftarrow \mathsf{Q_M} \cup \{(\mathsf{vkList}, \mathsf{M})\}$

$(\mathsf{pm}_1, \mathsf{st}_1) \xleftarrow{\$} \mathsf{Sign}_1^{(2)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, 1, \mathsf{sk})$

$\mathsf{Q_{st}}[\mathsf{sid}, 1] \leftarrow (\mathsf{vkList}, \mathsf{M}, \mathsf{st}_1)$

$\textbf{return } \mathsf{pm}_1$

$$\mathcal{O}_{\mathsf{Sign}_2^{(2)}}(\mathsf{sid}, (\mathsf{pm}_j)_{j \in [|\mathsf{vkList}|] \setminus \{1\}})$$

$\textbf{req } [\![\mathsf{Q_{st}}[\mathsf{sid}, 1] \neq \bot]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 2] = \bot]\!]$

$(\mathsf{vkList}, \mathsf{M}, \mathsf{st}_1) \leftarrow \mathsf{Q_{st}}[\mathsf{sid}, 1]$

$\mathsf{psig}_1 \xleftarrow{\$} \mathsf{Sign}_2^{(2)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, 1, \mathsf{sk}, \mathsf{st}_1, (\mathsf{pm}_j)_{j \in [|\mathsf{vkList}|] \setminus \{1\}})$

$\mathsf{Q_{st}}[\mathsf{sid}, 2] \leftarrow \mathsf{psig}_1$

$\textbf{return } \mathsf{psig}_1$

Figure 2.7: Unforgeability game for a two-round scheme in the random oracle model, where H denotes the random oracle.

$\mathsf{Game}^{\mathsf{ms^{1\text{-}1}\text{-}uf}}_{\mathsf{MS^{(1\text{-}1)}},\mathcal{A}}(1^\lambda, N)$

$\mathsf{Q_M} \leftarrow \emptyset, \mathsf{Q_{st}}[\cdot] \leftarrow \bot$

$\mathsf{par} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$

$(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}(\mathsf{par})$

$(\mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*) \xleftarrow{\$} \mathcal{A}^{\mathsf{Sign}_0^{(1\text{-}1)}, \mathsf{Sign}_1^{(1\text{-}1)}, \mathsf{H}}(\mathsf{par}, \mathsf{pk})$

$\textbf{req}\ [\![\mathsf{pk} \in \mathsf{vkList}^*]\!] \wedge [\![|\mathsf{vkList}^*| \leq N]\!] \wedge [\![(\mathsf{vkList}^*, \mathsf{M}^*) \notin \mathsf{Q_M}]\!]$

$\textbf{return}\ \mathsf{Verify}(\mathsf{par}, \mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*)$

$\mathsf{Sign}_0^{(1\text{-}1)}(\mathsf{sid})$

$\textbf{req}\ [\![\mathsf{Q_{st}}[\mathsf{sid}, 0] = \bot]\!]$

$(\mathsf{pm}_1, \mathsf{st}_1) \xleftarrow{\$} \mathsf{Sign}_0^{(1\text{-}1)}(\mathsf{par}, 1, \mathsf{sk})$

$\mathsf{Q_{st}}[\mathsf{sid}, 0] \leftarrow \mathsf{st}_1$

$\textbf{return}\ \mathsf{pm}_1$

$\mathsf{Sign}_1^{(1\text{-}1)}(\mathsf{sid}, \mathsf{vkList}, \mathsf{M}, (\mathsf{pm}_j)_{j \in [|\mathsf{vkList}|] \setminus \{1\}})$

$\textbf{req}\ [\![\mathsf{pk} \in \mathsf{vkList}]\!] \wedge [\![|\mathsf{vkList}| \leq N]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 0] \neq \bot]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 1] = \bot]\!]$

$\mathsf{Q_M} \leftarrow \mathsf{Q_M} \cup \{(\mathsf{vkList}, \mathsf{M})\}$

$(\mathsf{vkList}, \mathsf{M}, \mathsf{st}_1) \leftarrow \mathsf{Q_{st}}[\mathsf{sid}, 0]$

$\mathsf{psig}_1 \xleftarrow{\$} \mathsf{Sign}_1^{(1\text{-}1)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, 1, \mathsf{sk}, \mathsf{st}_1, (\mathsf{pm}_j)_{j \in |\mathsf{vkList}| \setminus \{1\}})$

$\mathsf{Q_{st}}[\mathsf{sid}, 1] \leftarrow \mathsf{psig}_1$

$\textbf{return}\ \mathsf{psig}_1$

Figure 2.8: Unforgeability game for a one-round scheme with pre-processing in the random oracle model, where H denotes the random oracle.

$\mathsf{Game}^{\mathsf{ms^3\text{-}uf}}_{\mathsf{MS}^{(3)},\mathcal{A}}(1^\lambda, N)$

$\mathsf{Q_M} \leftarrow \emptyset, \mathsf{Q_{st}}[\cdot] \leftarrow \bot$

$\mathsf{par} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$

$(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}(\mathsf{par})$

$(\mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}_1^{(3)}}, \mathcal{O}_{\mathsf{Sign}_2^{(3)}}, \mathcal{O}_{\mathsf{Sign}_3^{(3)}}, \mathsf{H}}(\mathsf{par}, \mathsf{pk})$

$\textbf{req } [\![\mathsf{pk} \in \mathsf{vkList}^*]\!] \wedge [\![|\mathsf{vkList}^*| \leq N]\!] \wedge [\![(\mathsf{vkList}^*, \mathsf{M}^*) \notin \mathsf{Q_M}]\!]$

$\textbf{return } \mathsf{Verify}(\mathsf{par}, \mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*)$

$\mathcal{O}_{\mathsf{Sign}_1^{(3)}}(\mathsf{sid}, \mathsf{vkList}, \mathsf{M})$

$\textbf{req } [\![\mathsf{pk} \in \mathsf{vkList}]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 1] = \bot]\!] \wedge [\![|\mathsf{vkList}| \leq N]\!]$

$\mathsf{Q_M} \leftarrow \mathsf{Q_M} \cup \{(\mathsf{vkList}, \mathsf{M})\}$

$(\mathsf{pm}_{1,1}, \mathsf{st}_1) \xleftarrow{\$} \mathsf{Sign}_1^{(3)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, 1, \mathsf{sk})$

$\mathsf{Q_{st}}[\mathsf{sid}, 1] \leftarrow (\mathsf{vkList}, \mathsf{M}, \mathsf{st}_1)$

$\textbf{return } \mathsf{pm}_{1,1}$

$\mathcal{O}_{\mathsf{Sign}_2^{(3)}}(\mathsf{sid}, (\mathsf{pm}_{1,j})_{j \in [|\mathsf{vkList}|] \setminus \{1\}})$

$\textbf{req } [\![\mathsf{Q_{st}}[\mathsf{sid}, 1] \neq \bot]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 2] = \bot]\!]$

$(\mathsf{vkList}, \mathsf{M}, \mathsf{st}_1) \leftarrow \mathsf{Q_{st}}[\mathsf{sid}, 1]$

$(\mathsf{pm}_{2,1}, \mathsf{st}_1) \xleftarrow{\$} \mathsf{Sign}_2^{(3)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, 1, \mathsf{sk}, \mathsf{st}_1, (\mathsf{pm}_{1,j})_{j \in [N] \setminus \{1\}})$

$\mathsf{Q_{st}}[\mathsf{sid}, 2] \leftarrow (\mathsf{vkList}, \mathsf{M}, \mathsf{st}_1)$

$\textbf{return } \mathsf{pm}_{2,1}$

$\mathcal{O}_{\mathsf{Sign}_3^{(3)}}(\mathsf{sid}, (\mathsf{pm}_{1,j})_{j \in [|\mathsf{vkList}|] \setminus \{1\}})$

$\textbf{req } [\![\mathsf{Q_{st}}[\mathsf{sid}, 2] \neq \bot]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 3] = \bot]\!]$

$(\mathsf{vkList}, \mathsf{M}, \mathsf{st}_1) \leftarrow \mathsf{Q_{st}}[\mathsf{sid}, 2]$

$\mathsf{psig}_1 \xleftarrow{\$} \mathsf{Sign}_3^{(3)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, 1, \mathsf{sk}, \mathsf{st}_1, (\mathsf{pm}_{2,j})_{j \in [|\mathsf{vkList}|] \setminus \{1\}})$

$\mathsf{Q_{st}}[\mathsf{sid}, 3] \leftarrow \mathsf{psig}_1$

$\textbf{return } \mathsf{psig}_1$

Figure 2.9: Unforgeability game for a three-round scheme in the random oracle model, where H denotes the random oracle.

$\mathsf{Game}^{\mathsf{ms^2\text{-}uf1}}_{\mathsf{MS}^{(2)},\mathcal{A}}(1^\lambda, N)$, $\boxed{\mathsf{Game}^{\mathsf{ms^2\text{-}uf2}}_{\mathsf{MS}^{(2)},\mathcal{A}}(1^\lambda, N)}$

$\mathsf{Q_M} \leftarrow \emptyset, \mathsf{Q_{st}}[\cdot] \leftarrow \bot$

$\mathsf{par} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$

$(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}(\mathsf{par})$

$(\mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}_1^{(2)}}, \mathcal{O}_{\mathsf{Sign}_2^{(2)}}, \mathsf{H}}(\mathsf{par}, \mathsf{pk})$

$\mathbf{req}\ [\![\mathsf{pk} \in \mathsf{vkList}^*]\!] \wedge [\![|\mathsf{vkList}^*| \leq N]\!] \wedge [\![(\mathsf{vkList}^*, \mathsf{M}^*) \notin \mathsf{Q_M}]\!]$ // For $\mathsf{Game}^{\mathsf{ms^2\text{-}uf1}}_{\mathsf{MS}^{(2)},\mathcal{A}}$.

$\boxed{\mathbf{req}\ [\![\mathsf{pk} \in \mathsf{vkList}^*]\!] \wedge [\![|\mathsf{vkList}^*| \leq N]\!] \wedge [\![\mathsf{M}^* \notin \mathsf{Q_M}]\!]}$ // For $\mathsf{Game}^{\mathsf{ms^2\text{-}uf2}}_{\mathsf{MS}^{(2)},\mathcal{A}}$.

$\mathbf{return}\ \mathsf{Verify}(\mathsf{par}, \mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*)$

$\mathcal{O}_{\mathsf{Sign}_1^{(2)}}(\mathsf{sid}, \mathsf{vkList}, \mathsf{M})$

$\mathbf{req}\ [\![\mathsf{pk} \in \mathsf{vkList}]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 1] = \bot]\!] \wedge [\![|\mathsf{vkList}| \leq N]\!]$

$\mathsf{HS_{sid}} \leftarrow \emptyset$

$\mathbf{parse}\ (\mathsf{pk}_i)_{i \in [|\mathsf{vkList}|]} \leftarrow \mathsf{vkList}$

$\mathbf{for}\ i \in [|\mathsf{vkList}|]\ \mathbf{do}$

$\quad \mathbf{if}\ \mathsf{pk}_i = \mathsf{pk}\ \mathbf{then}$

$\quad\quad \mathsf{HS_{sid}} \leftarrow \mathsf{HS_{sid}} \cup \{i\}$

$\mathbf{for}\ i \in \mathsf{HS_{sid}}\ \mathbf{do}\ (\mathsf{pm}_i, \mathsf{st}_i) \xleftarrow{\$} \mathsf{Sign}_1^{(2)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk})$

$\mathsf{Q_{st}}[\mathsf{sid}, 1] \leftarrow (\mathsf{vkList}, \mathsf{M}, \mathsf{HS_{sid}}, (\mathsf{st}_i)_{i \in \mathsf{HS_{sid}}})$

$\mathbf{return}\ (\mathsf{pm}_i)_{i \in \mathsf{HS_{sid}}}$

$\mathcal{O}_{\mathsf{Sign}_2^{(2)}}(\mathsf{sid}, (\mathsf{pm}_j)_{j \in [|\mathsf{vkList}|] \setminus \mathsf{HS}})$

$\mathbf{req}\ [\![\mathsf{Q_{st}}[\mathsf{sid}, 1] \neq \bot]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 2] = \bot]\!]$

$(\mathsf{vkList}, \mathsf{M}, \mathsf{HS_{sid}}, (\mathsf{st}_i)_{i \in \mathsf{HS_{sid}}}) \leftarrow \mathsf{Q_{st}}[\mathsf{sid}, 1]$

$\mathbf{for}\ i \in \mathsf{HS_{sid}}\ \mathbf{do}\ \mathsf{psig}_i \xleftarrow{\$} \mathsf{Sign}_2^{(2)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}, \mathsf{st}_i, (\mathsf{pm}_j)_{j \in [|\mathsf{vkList}|] \setminus \{i\}})$

$\mathsf{Q_{st}}[\mathsf{sid}, 2] \leftarrow (\mathsf{psig}_i)_{i \in \mathsf{HS_{sid}}}$

$\mathsf{Q_M} \leftarrow \mathsf{Q_M} \cup \{(\mathsf{vkList}, \mathsf{M})\}$ // For $\mathsf{Game}^{\mathsf{ms^2\text{-}uf1}}_{\mathsf{MS}^{(2)},\mathcal{A}}$.

$\boxed{\mathsf{Q_M} \leftarrow \mathsf{Q_M} \cup \{\mathsf{M}\}}$ // For $\mathsf{Game}^{\mathsf{ms^2\text{-}uf2}}_{\mathsf{MS}^{(2)},\mathcal{A}}$.

$\mathbf{return}\ (\mathsf{psig}_i)_{i \in \mathsf{HS_{sid}}}$

Figure 2.10: Slightly strong and weak unforgeability games for a two-round schemes in the random oracle model, where H denotes the random oracle.

Chapter 3

Discrete-Logarithm-Based Multi-Signatures

In this chapter, we review DL-based multi-signature schemes. From the first multi-signature scheme proposed by [IN83] to this day, many DL-based multi-signature schemes have been proposed. Let us roughly look back at this long history of the DL-based multi-signature schemes. We divide this long history into three periods by focusing on the two papers [BN06] and [DEF+19]. In the first paper [BN06], Bellare and Neven proposed the first three-round multi-signature scheme in the *plain public-key (PPK) model*. The second paper by Drijvers et al . suggested the vulnerability of two-round multi-signature schemes proposed so far and proposed the first secure two-round multi-signature scheme in the PPK model.

Most of the early proposed schemes [IN83, LHL95, Lan96, MH96, OO93, OO99, Har94] before the first paper appeared are insecure against the rogue-key attack. In this attack, an adversary maliciously generates cosigners' public keys to forge. This emphasizes the importance of *the key setup*. A naive approach to prevent this attack is to append the certification of knowledge of the secret key to the public key. This restriction is modeled as the key verification model [BJ08] or the proof-of-possession model [RY07]. As a more theoretical model, there exists the knowledge of secret key model [Bol03, LOS+06]. In the unforgeability game under such key setup models, a challenger forces an adversary to output not only cosigners' public keys but also the corresponding secret keys or certifications of knowledge of secret keys. Another approach is the dedicated key generation introduced by [MOR01], in which all potential signers execute the interactive protocol to generate each public and secret key. While this approach prevents the attack, it makes schemes not practical. We will provide the details on the rogue-key attack and the key setup models in Section 3.2.

Bellare and Neven proposed the first multi-signature scheme that is secure against the rouge-key attack without impractical interaction protocols and assuming restricted key setup models [BN06]. This unrestricted key setup model is called the plain public-key (PPK) model, in which an adversary is allowed to generate cosigners' public keys freely. Moreover, the multi-signature of this scheme consists of one group element and one scalar as same as the Schnorr signature. After this scheme was proposed, the main research direction is to reduce the number of rounds of the signing protocol, namely, to construct a two-round multi-signature scheme. Moreover, Maxwell et al. introduced the notion of the key aggregation [MPSW19]. This property provides more efficient verification and is significant in the application of cryptocurrencies. Within this context, the primary desirable features of multi-signature schemes are the security in the PPK model, supporting key aggregation, and the two-round signing protocol.

Although some two-round schemes [BCJ08, MWLD10, STV+16, MPSW18] have been proposed since the Bellare-Neven scheme was proposed, Drijvers et al. suggested that they are insecure [DEF+19]. Indeed, they showed the sub-exponential attack based on Wagner's k-sum algorithm [Wag02] exploiting the concurrent signing session. Subsequently, Benhamouda et al. proposed a more efficient attack of polynomial complexity under a certain condition, a.k.a, the ROS attack [BLL+21]. Drijvers et al. also proposed the first two-round multi-signature scheme that is secure in the PPK model. After that, several two-round schemes are constructed achieving the above-mentioned desirable properties.

Road Maps. In Section 3.1, we first review the Schnorr signature scheme on which most DL-based multi-signature schemes are based. In Section 3.2, we explain the rogue-key attack and the restricted key setup model. In Section 3.3, we show the constructions and the security theorems of some three-round multi-signature schemes. Finally, we show the constructions and the security theorems of some two-round multi-signature schemes in Section 3.4.

3.1 Schnorr Signature Scheme

In this section, we review the Schnorr signature scheme [Sch90]. The construction of this scheme is shown in Section 3.1.

The Schnorr signature scheme is proven secure under the discrete logarithm (DL) assumption and the random oracle model. We briefly explain how the security of this is proven. We construct an adversary $\mathcal{B}$ against the DL problem that internally runs an adversary $\mathcal{A}$ against the scheme. $\mathcal{B}$

$\mathsf{KeyGen}(1^\lambda)$	$\mathsf{Sig}(\mathsf{sk}, \mathsf{M})$	$\mathsf{Vf}(\mathsf{pk}, \mathsf{M}, \mathsf{sig})$
$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$	$r \xleftarrow{\$} \mathbb{Z}_p$	**parse** $(c, s) \leftarrow \mathsf{Sig}$
$x \xleftarrow{\$} \mathbb{Z}_p$	$R \leftarrow rG$	$R \leftarrow sG - cX$
$X \leftarrow xG$	$c \leftarrow \mathsf{H}(R, \mathsf{M})$	**return** $(c = \mathsf{H}(R, \mathsf{M}))$
$\mathsf{sk} \leftarrow x$	$s \leftarrow r + cx \mod p$	
$\mathsf{pk} \leftarrow X$	$\mathsf{sig} \leftarrow (c, s)$	
return $(\mathsf{pk}, \mathsf{sk})$	**return** sig	

Figure 3.1: The Schnorr signature scheme. H is a hash function $\mathsf{H} : \{0,1\}^* \rightarrow \mathbb{Z}_p$ modeled as a random oracle.

takes as input X, which is an instance of the DL problem, and embeds X into the challenge public key pk. It runs $\mathcal{A}$ by giving pk and answering signing queries. It responds to signing queries by exploiting the random oracle and the honest verifier zero-knowledge (HVZK). Specifically, it first chooses $s, c \xleftarrow{\$} \mathbb{Z}_p$ and computes $R \leftarrow sG - cX$. Before returning (s, c) to $\mathcal{A}$, it assigns $\mathsf{H}(R, \mathsf{M}) \leftarrow c$ in the random oracle. Eventually, $\mathcal{A}$ outputs a forgery (s_1^*, c_1^*) on a message M^*, which has never appeared in the signing queries. Then, it rewinds $\mathcal{A}$ at the point when $\mathsf{H}(s_1^*G - c_1^*X, \mathsf{M}^*)$ is queried to the random oracle, re-defines $\mathsf{H}(s_1^*G - c_1^*X, \mathsf{M}^*) \leftarrow c_2^*$, and runs $\mathcal{A}$ again. If $\mathcal{A}$ outputs a forgery (s_2^*, c_2^*) under the same $\mathsf{H}(s_1^*G - c_1^*X, \mathsf{M}^*)$, it obtains (s_1^*, c_1^*) and (s_2^*, c_2^*) such that $s_1^*G - c_1^*X = s_2^*G - c_2^*X$ and then can extract the discrete log of X as $(s_1^* - s_2^*)/(c_1^* - c_2^*)$ when $c_1^* \neq c_2^*$. The success probability of $\mathcal{B}$ can be evaluated by using the forking lemma [PS00, BN06]. Consequently, it has the reduction loss $O(Q_H \epsilon)$ where Q_H is the number of the random oracle queries and ϵ is the advantage of $\mathcal{A}$. On the other hand, additionally assuming the algebraic group model, we can prove that the Schnorr signature scheme is tightly secure[FPS20].

3.2 Rogue Key Attack and Restricted Key Setup Models

In order to demonstrate the rogue-key attack, we consider an insecure multi-signature scheme based on the Schnorr signature scheme. The secret and public keys are the same as those of the Schnorr signature scheme, namely, $\mathsf{sk} = x \in \mathbb{Z}_p$ and $\mathsf{pk} = X = xG \in \mathbb{G}$, respectively. In the signing protocol, each signer first generates R_i and broadcasts it to other cosigners. After receiving $(R_i)_i$, it aggregates them into $\widetilde{R} = \sum_i R_i$, computes $c \leftarrow \mathsf{H}(\widetilde{R}, \mathsf{M})$

and $s \leftarrow r_i + c_i x_i \mod p$ and broadcasts s_i. Finally, it obtains $(s_i)_i$ and produces the multi-signature $(c, \widetilde{s})$ on M by computing $\widetilde{s} = \sum_i s_i$. The multi-signature is verified by computing $\widetilde{R} \leftarrow \widetilde{s}G - c\sum_i X_i$ and checking $c = \mathsf{H}(\widetilde{R}, \mathsf{M})$.

In the rogue-key attack, an adversary maliciously generates cosigners' public keys to forge the multi-signature. The attack against the above insecure scheme is quite simple. Let X_1 be an honest signer's public key. The adversary produces the cosigner's public key by $X_2 \leftarrow x_2 G - X$ where $x_2 \in \mathbb{Z}_p$. Then, since $X_1 + X_2 = x_2 G$ holds, it can easily generate a forgery on any message. One may wonder if we can prevent the rogue-key attack if the hash function additionally takes as input the public key, e.g., $\mathsf{H}(\widetilde{R}, \mathsf{M}, \mathsf{pk}_i)$. Indeed, X_1 is not canceled out by $c_1 X_1 + c_2 X_2$ where $c_i = \mathsf{H}(\widetilde{R}, \mathsf{M}, X_i)$ for $i \in \{1, 2\}$. However, the modified scheme is also insecure. Specifically, the adversary embeds X_1 in all cosigners' public keys as $X_i \leftarrow x_i G + a_i X_1$ where $x_i, a_i \in \mathbb{Z}_p$. It can make many $c_i = \mathsf{H}(\widetilde{R}, \mathsf{M}, X_i)$ by re-choosing x_i and then it can find $(c_i)_i$ satisfying $\sum_i c_i a_i \equiv 0 \mod p$. Then, $\sum_i c_i X_i = \sum_i c_i x_i G + \sum_i c_i a_i X_i = \sum_i c_i x_i G$ holds where $x_1 = 0$ and $a_1 = 1$. Since an adversary knows $(x_i)_i$, it can forge on any message.

A naive way to prevent the rogue-key attack is that each signer issues a certification of knowledge of the secret key together with the public key. If the adversary generates cosigners' public keys from an honest signer's key, it cannot generate such certifications since it does not know the secret keys. This approach is effective in some applications, in which a trusted key registration server exists. On the other hand, in the case where the existence of such a trusted party is not guaranteed, the approach makes the multi-signature scheme meaningless. In such a case, the verifier needs to verify not only a multi-signature but also certifications. This means that the multi-signature scheme is as inefficient as simply concatenating individual signatures.

Another approach is the dedicated key generation introduced by [MOR01]. The set of potential signers executes the interactive key generation protocol, as a pre-processing. While this prevents the rogue-key attack, the signer set is necessarily static. Whereas this is a good solution in static applications, this approach is not suitable for dynamic situations. Moreover, this approach obviously makes schemes inefficient.

The key verification (KV) model [BJ08] and proof-of-possession (PoP) model [RY07] are the restricted key setup models that capture the first naive approach. In these models, while an adversary is allowed to generate cosigners' public keys, it needs to issue the certifications of knowledge of corresponding secret keys. If the certifications are invalid, it is defeated in the unforgeability game. As a more theoretical model, there exists the knowledge of secret key (KOSK) model [Bol03, LOS+06]. In this model, an adversary needs to output cosigners' secret keys instead of the certifications. This model is theoretical since issuing a secret key is unrealistic in practice.

Bellare-Neven multi-signature scheme is the first secure DL-based multi-signature scheme without restricted key setup models. This key setup model is called the plain public key (PPK) model. We will review the construction of this scheme in Section 3.3. Here, we explain the intuition of how it prevents the rogue-key attack. In this scheme, the challenge c is generated by the hash function $\mathsf{H}(\tilde{R}, \mathsf{M}, \mathsf{vkList}, \mathsf{pk}_i)$ where vkList is the list of public keys. We notice that $(c_i)_i$ is completely linked to all signers' public keys. In other words, when one replaces a key with a different key, all values in $(c_i)_i$ are changed with overwhelming probability since the inputs of the hash are changed. This means that any adversary can no longer find $(c_i)_i$ canceling out the public key of the honest signer except for a negligible probability.

3.3 Three-Round DL-Based Multi-Signatures

In this section, we survey some three-round multi-signature schemes. Specifically, we show the constructions of the Bellare-Neven scheme and $\mathsf{MuSig\text{-}DL}$ and restate the security theorems for them. Note that we restate them under a common notation and definition in Chapter 2 to help with comparisons. Moreover, since the security of all schemes is proven in the random oracle model and the PPK model, we do not mentioned them explicitly.

3.3.1 Bellare-Neven Scheme

Bellare and Neven proposed the first DL-based three-round multi-signature scheme $\mathsf{BN\text{-}DL}$ which is proven secure under the DL assumption. We show the construction in Fig. 3.2. In the document [BN05], they also proposed a DDH-based scheme which is built from the tightly secure signature scheme, i.e., the Katz-Wang signature scheme.

We restate the security theorem below.

Theorem 1 ([BN06, Theorem 4]). *If there exists an adversary $\mathcal{A}$ that $(t, Q_S, Q_H, N, \epsilon)$-breaks the 3-MS-UF of* $\mathsf{BN\text{-}DL}$*, then there exists an adversary $\mathcal{B}$ that (t', ϵ')-solves the DL problem such that*

$$\epsilon' \geq \frac{\epsilon^2}{Q_H + Q_S} - \frac{2Q_H + 16N^2 Q_S}{2^\ell} - \frac{8NQ_S + 1}{p},$$
$$t' = 2t + Q_S t_{\mathrm{mul}} + O((Q_H + Q_S)(Q_H + NQ_S + 1)),$$

where t_{mul} is the time of an scalar multiplication in $\mathbb{G}$.

$\mathsf{Setup}(1^\lambda)$:

$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$
Select $\mathsf{H_c}$ // $\mathsf{H_c} : \{0,1\}^* \to \mathbb{Z}_p$
Select $\mathsf{H_{com}}$ // $\mathsf{H_{com}} : \{0,1\}^* \to \{0,1\}^\ell$
return $\mathsf{par} = (\mathbb{G}, p, G, \mathsf{H_c}, \mathsf{H_{com}})$.

$\mathsf{KeyGen}(\mathsf{par}) \to (\mathsf{pk}, \mathsf{sk})$:

$x \xleftarrow{\$} \mathbb{Z}_p$
$X \leftarrow xG$
$\mathsf{pk} \leftarrow X$
$\mathsf{sk} \leftarrow x$
return $(\mathsf{pk}, \mathsf{sk})$

$\mathsf{Agg}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, (\mathsf{pm}_i, \mathsf{psig}_i)_{i\in[N]})$:

parse $(t_i, R_i, s_i)_{i\in[N]} \leftarrow (\mathsf{pm}_i, \mathsf{psig}_i)_{i\in[N]}$
$\widetilde{R} \leftarrow \sum_{i=1}^{N} R_i$
$\widetilde{s} \leftarrow \sum_{i=1}^{N} s_i$
$\widetilde{\mathsf{sig}} \leftarrow (\widetilde{R}, \widetilde{s})$
return $\widetilde{\mathsf{sig}}$

$\mathsf{Verify}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, \widetilde{\mathsf{sig}})$:

parse $(X_i)_{i\in[N]} \leftarrow \mathsf{vkList}$
parse $(\widetilde{R}, \widetilde{s}) \leftarrow \widetilde{\mathsf{sig}}$
for $i \in [N]$ **do**
  $c_i \leftarrow \mathsf{H_c}(\widetilde{R}, \mathsf{M}, \mathsf{vkList}, X_i)$
if $\left[\!\left[\widetilde{R} = \widetilde{s}G - \sum_{i=1}^{N} c_i X_i\right]\!\right]$ **then**
  return 1
return 0

$\mathsf{Sign}_1^{(3)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i)$:

$r_i \xleftarrow{\$} \mathbb{Z}_p$
$R_i \leftarrow r_i G$
$t_i \leftarrow \mathsf{H_{com}}(R_i)$
$\mathsf{st}_i \leftarrow (r_i, R_i, t_i)$
$\mathsf{pm}_{1,i} \leftarrow t_i$
return $(\mathsf{pm}_{1,i}, \mathsf{st}_i)$

$\mathsf{Sign}_2^{(3)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_{1,j})_{j\in[N]\setminus\{i\}})$:

parse $(t_j)_{j\in[N]\setminus\{i\}} \leftarrow (\mathsf{pm}_{1,j})_{j\in[N]\setminus\{i\}}$
parse $(r_i, R_i, t_i) \leftarrow \mathsf{st}_i$
$\mathsf{st}_i \leftarrow \mathsf{st}_i \cup \{(t_j)_{j\in[N]\setminus\{i\}}\}$
$\mathsf{pm}_{2,i} \leftarrow R_i$
return $(\mathsf{pm}_{2,i}, \mathsf{st}_i)$

$\mathsf{Sign}_3^{(3)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_{2,j})_{j\in[N]\setminus\{i\}})$:

parse $(R_j)_{j\in[N]\setminus\{i\}} \leftarrow (\mathsf{pm}_{2,j})_{j\in[N]\setminus\{i\}})$
parse $(r_i, R_i, t_i, (t_j)_{j\in[N]\setminus\{i\}}) \leftarrow \mathsf{st}_i$
req $\left[\!\left[{}^\forall i \in [N],\ t_i = \mathsf{H_{com}}(R_i)\right]\!\right]$
$\widetilde{R} \leftarrow \sum_{i=1}^{N} R_i$
$c_i \leftarrow \mathsf{H_c}(\widetilde{R}, \mathsf{M}, \mathsf{vkList}, X_i)$
$s_i \leftarrow x_i c_i + r_i \mod p$
$\mathsf{psig}_i \leftarrow s_i$
return psig_i

Figure 3.2: The construction of BN-DL. $\mathsf{H_c}$ and $\mathsf{H_{com}}$ are modeled as random oracles. N is the number of the signers in vkList.

3.3.2 MuSig-DL

Boneh et al. [BDN18] and Maxwell et al. [MPSW19] proposed a variant scheme MuSig-DL of BN-DL that supports key aggregation [MPSW19]. The construction of MuSig-DL is similar to that of BN-DL as shown in Fig. 3.3.

$\mathsf{Setup}(1^\lambda)$:

- $(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$
- **Select** $\mathsf{H_c}$ // $\mathsf{H_c} : \{0,1\}^* \to \{0,1\}^\ell$
- **Select** $\mathsf{H_{com}}$ // $\mathsf{H_{com}} : \{0,1\}^* \to \{0,1\}^\ell$
- **Select** $\mathsf{H_{agg}}$ // $\mathsf{H_{agg}} : \{0,1\}^* \to \{0,1\}^\ell$
- **return** $\mathsf{par} = (\mathbb{G}, p, G, \mathsf{H_c}, \mathsf{H_{com}}, \mathsf{H_{agg}})$.

$\mathsf{KeyGen}(\mathsf{par}) \to (\mathsf{pk}, \mathsf{sk})$:

- $x \xleftarrow{\$} \mathbb{Z}_p$
- $X \leftarrow xG$
- $\mathsf{pk} \leftarrow X$
- $\mathsf{sk} \leftarrow x$
- **return** $(\mathsf{pk}, \mathsf{sk})$

$\mathsf{Agg}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, (\mathsf{pm}_i, \mathsf{psig}_i)_{i\in[N]})$:

- **parse** $(t_i, R_i, s_i)_{i\in[N]} \leftarrow (\mathsf{pm}_i, \mathsf{psig}_i)_{i\in[N]}$
- $\widetilde{R} \leftarrow \sum_{i=1}^{N} R_i$
- $\widetilde{s} \leftarrow \sum_{i=1}^{N} s_i$
- $\widetilde{\mathsf{sig}} \leftarrow (\widetilde{R}, \widetilde{s})$
- **return** $\widetilde{\mathsf{sig}}$

$\mathsf{Verify}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, \widetilde{\mathsf{sig}})$:

- **parse** $(X_i)_{i\in[N]} \leftarrow \mathsf{vkList}$
- **parse** $(\widetilde{R}, \widetilde{s}) \leftarrow \widetilde{\mathsf{sig}}$
- **for** $i \in [N]$ **do**
  - $a_i \leftarrow \mathsf{H_{agg}}(X_i, \mathsf{vkList})$
- $\widetilde{X} = \sum_{i=1}^{N} a_i X_i$
- $c \leftarrow \mathsf{H_c}(\widetilde{R}, \mathsf{M}, \widetilde{X})$
- **if** $\left[\!\left[\widetilde{R} = \widetilde{s}G - c\widetilde{X}\right]\!\right]$ **then**
  - **return** 1
- **return** 0

$\mathsf{Sign}_1^{(3)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i)$:

- $r_i \xleftarrow{\$} \mathbb{Z}_p$
- $R_i \leftarrow r_i G$
- $t_i \leftarrow \mathsf{H_{com}}(R_i)$
- $\mathsf{st}_i \leftarrow (r_i, R_i, t_i)$
- $\mathsf{pm}_{1,i} \leftarrow t_i$
- **return** $(\mathsf{pm}_{1,i}, \mathsf{st}_i)$

$\mathsf{Sign}_2^{(3)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_{1,j})_{j\in[N]\setminus\{i\}})$:

- **parse** $(t_j)_{j\in[N]\setminus\{i\}} \leftarrow (\mathsf{pm}_{1,j})_{j\in[N]\setminus\{i\}}$
- **parse** $(r_i, R_i, t_i) \leftarrow \mathsf{st}_i$
- $\mathsf{st}_i \leftarrow \mathsf{st}_i \cup \{(t_j)_{j\in[N]\setminus\{i\}}\}$
- $\mathsf{pm}_{2,i} \leftarrow R_i$
- **return** $(\mathsf{pm}_{2,i}, \mathsf{st}_i)$

$\mathsf{Sign}_3^{(3)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_{2,j})_{j\in[N]\setminus\{i\}})$:

- **parse** $(R_j)_{j\in[N]\setminus\{i\}} \leftarrow (\mathsf{pm}_{2,j})_{j\in[N]\setminus\{i\}})$
- **parse** $(r_i, R_i, t_i, (t_j)_{j\in[N]\setminus\{i\}}) \leftarrow \mathsf{st}_i$
- **parse** $(X_i)_{i\in[N]} \leftarrow \mathsf{vkList}$
- **req** $\left[\!\left[\forall i \in [N],\ t_i = \mathsf{H_{com}}(R_i)\right]\!\right]$
- **for** $i \in [N]$ **do**
  - $a_i \leftarrow \mathsf{H_{agg}}(X_i, \mathsf{vkList})$
- $\widetilde{X} = \sum_{i=1}^{N} a_i X_i$
- $\widetilde{R} \leftarrow \sum_{i=1}^{N} R_i$
- $c \leftarrow \mathsf{H_c}(\widetilde{R}, \mathsf{M}, \widetilde{X})$
- $s_i \leftarrow x_i a_i c + r_i \mod p$
- $\mathsf{psig}_i \leftarrow s_i$
- **return** psig_i

Figure 3.3: The construction of MuSig-DL. $\mathsf{H_c}, \mathsf{H_{com}}$ and $\mathsf{H_{agg}}$ are modeled as random oracles. N is the number of the signers in vkList.

In [FH21], a variant scheme of MuSig-DL that is proven tightly secure under the DDH assumption is proposed. Below, we only restate the security theorem of MuSig-DL.

Theorem 2 ([MPSW19, Theorem 1]). *If there exists an adversary $\mathcal{A}$ that $(t, Q_S, Q_H, N, \epsilon)$-breaks the 3-MS-UF of* MuSig-DL*, then there exists an adversary $\mathcal{B}$ that (t', ϵ')-solves the DL problem such that*

$$\epsilon' \geq \frac{\epsilon^4}{(Q_H + Q_S + 1)^3} - \frac{16Q_S(Q_H + NQ_S)}{p} - \frac{16(Q_H + NQ_S)^2 + 3}{2^\ell},$$
$$t' = 4t + 4Nt_{\text{mul}} + (N(Q_H + Q_S + 1)),$$

where t_{mul} is the time of an scalar multiplication in $\mathbb{G}$.

3.4 Two-Round DL-Based Multi-Signatures

In this section, we survey two-round signature schemes. Specifically, we show the constructions of schemes and restate the security theorems. Note that we restate them under a common notation and definition in Chapter 2 to help with comparisons. Moreover, unless noted otherwise, the security of schemes is proven in the random oracle model and the PPK model.

3.4.1 Modified BCJ

Drijvers et al. proposed a first secure two-round multi-signature scheme mBCJ-KV [DEF+19]. This scheme was constructed by applying the patch to the insecure two-round multi-signature scheme BCJ [BCJ08]. They showed the construction and proved that mBCJ-KV is secure under the DL assumption *in the key verification model*. Moreover, they roughly described how to modify it to be secure in the PPK model. However, there is no concrete construction and no formal security proof. Here, we show a variant scheme of mBCJ which is secure in the PPK model by applying the way. We call this scheme as mBCJ-PPK. We show the construction of mBCJ-PPK in Fig. 3.4.

The following theorem states that mBCJ-PPK is secure under the DL assumption in the PPK model. We will prove this theorem in Section 3.5.

Theorem 3. *If there exists an adversary $\mathcal{A}$ that $(t_{\mathcal{A}}, Q_S, Q_H, N, \epsilon_{\mathcal{A}})$-breaks the 2-MS-UF of* mBCJ-PPK*, then there exists an adversary $\mathcal{B}$ that $(t_{\mathcal{B}}, \epsilon_{\mathcal{B}})$-solves the DL problem such that*

$$\epsilon_{\mathcal{B}} \geq \left(1 - \frac{Q_S}{p}\right)^2 \frac{\epsilon_{\mathcal{A}}^2}{N(Q_H + Q_S + 1)e^2(Q_S + 1)^2} - \frac{1}{p}, \quad \text{and}$$

$\mathsf{Setup}(1^\lambda)$:

$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$
Select $\mathsf{H_c}$ // $\mathsf{H_c} : \{0,1\}^* \to \mathbb{Z}_p$
Select $\mathsf{H_{ck}}$ // $\mathsf{H_{ck}} : \{0,1\}^* \to \mathbb{G}^3$
return $\mathsf{par} = (\mathbb{G}, p, G, \mathsf{H_c}, \mathsf{H_{ck}})$.

$\mathsf{KeyGen}(\mathsf{par}) \to (\mathsf{pk}, \mathsf{sk})$:

$x \xleftarrow{\$} \mathbb{Z}_p$
$X \leftarrow xG$,
$\mathsf{pk} \leftarrow X$
$\mathsf{sk} \leftarrow x$
return $(\mathsf{pk}, \mathsf{sk})$

$\mathsf{Agg}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, (\mathsf{pm}_i, \mathsf{psig}_i)_{i\in[N]})$:

parse $(X_i)_{i\in[N]} \leftarrow \mathsf{vkList}$
parse $(T_{i,1}, T_{i,2}, s_i, \alpha_i, \beta_i)_{i\in[N]} \leftarrow (\mathsf{pm}_i, \mathsf{psig}_i)_{i\in[N]}$
$\widetilde{T}_1 \leftarrow \sum_{i=1}^{N} T_{i,1}$
$\widetilde{T}_2 \leftarrow \sum_{i=1}^{N} T_{i,2}$
for $i \in [N]$ **do**
  $c_i \leftarrow \mathsf{H_c}(X_i, \widetilde{T}_1, \widetilde{T}_2, \mathsf{vkList}, \mathsf{M})$
$\tilde{s} \leftarrow \sum_{i=1}^{N} s_i \mod p$
$\tilde{\alpha} \leftarrow \sum_{i=1}^{N} \alpha_i \mod p$
$\tilde{\beta} \leftarrow \sum_{i=1}^{N} \beta_i \mod p$
$\widetilde{R} \leftarrow \tilde{s}G - \sum_{i=1}^{N} c_i X_i$
$\widetilde{\mathsf{sig}} \leftarrow (\widetilde{R}, \tilde{s}, \tilde{\alpha}, \tilde{\beta})$
return $\widetilde{\mathsf{sig}}$

$\mathsf{Sign}_1^{(2)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i)$:

$(U_1, U_2, U_3) \leftarrow \mathsf{H_{ck}}(\mathsf{M}, \mathsf{vkList})$
$r_i, \alpha_i, \beta_i \xleftarrow{\$} \mathbb{Z}_p$
$T_{i,1} \leftarrow \alpha_i G + \beta_i U_2$
$T_{i,2} \leftarrow \alpha_i U_1 + \beta_i U_3 + r_i G$
$\mathsf{pm}_i \leftarrow (T_{i,1}, T_{i,2})$
$\mathsf{st}_i \leftarrow (r_i, \alpha_i, \beta_i, T_{i,1}, T_{i,2})$
return $(\mathsf{pm}_i, \mathsf{st}_i)$

$\mathsf{Sign}_2^{(2)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_j)_{j\in[N]\setminus\{i\}})$:

parse $(T_{j,1}, T_{j,2})_{j\in[N]\setminus\{i\}} \leftarrow (\mathsf{pm}_j)_{j\in[N]\setminus\{i\}}$
parse $(r_i, \alpha_i, \beta_i, T_{i,1}, T_{i,2}) \leftarrow \mathsf{st}_i$
$\widetilde{T}_1 \leftarrow \sum_{j=1}^{N} T_{j,1}$
$\widetilde{T}_2 \leftarrow \sum_{j=1}^{N} T_{j,2}$
$c_i \leftarrow \mathsf{H_c}(X_i, \widetilde{T}_1, \widetilde{T}_2, \mathsf{vkList}, \mathsf{M})$
$s_i \leftarrow x_i c_i + r_i \mod p$
$\mathsf{psig}_i \leftarrow (s_i, \alpha_i, \beta_i)$
return psig_i

$\mathsf{Verify}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, \widetilde{\mathsf{sig}})$:

parse $(X_i)_{i\in[N]} \leftarrow \mathsf{vkList}$
parse $(\widetilde{R}, \tilde{s}, \tilde{\alpha}, \tilde{\beta}) \leftarrow \widetilde{\mathsf{sig}}$
$(U_1, U_2, U_3) \leftarrow \mathsf{H_{ck}}(\mathsf{M}, \mathsf{vkList})$
$\widetilde{T}_1 \leftarrow \tilde{\alpha}G + \tilde{\beta}U_2$
$\widetilde{T}_2 \leftarrow \tilde{\alpha}U_1 + \tilde{\beta}U_3 + \widetilde{R}$
for $i \in [N]$ **do**
  $c_i \leftarrow \mathsf{H_c}(X_i, \widetilde{T}_1, \widetilde{T}_2, \mathsf{vkList}, \mathsf{M})$
if $\left[\!\left[\widetilde{R} = \tilde{s}G - \sum_{i=1}^{N} c_i X_i \right]\!\right]$ **then**
  return 1
return 0

Figure 3.4: The construction of mBCJ-PPK. $\mathsf{H_c}$ and $\mathsf{H_{ck}}$ are modeled as random oracles. N is the number of the signers in vkList.

$$t_{\mathcal{B}} \le 2t_{\mathcal{A}} + (6Q_H + 12Q_S + 2N + 16)t_{\mathrm{mul}} + O(N(Q_S + Q_H)),$$

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

3.4.2 MuSig-DN

MuSig-DN [NRSW20] is a two-round multi-signature scheme. To prevent the ROS attack (and the k-sum attack), all signers execute the signing protocol with deterministic nonces. This means that the multi-signature is determined by a public-key list and a message to be signed. This scheme achieves a two-round signing protocol by using pseudorandom functions (PRF), pseudorandom number generators (PRNG), and succinct non-interactive arguments of knowledge (SNARKs) [BBB+18] to make the signing protocol deterministic. Note that this scheme requires additionally one round when all signers who participate in the signing protocol do not share the keys deterministically computed from the secret key by using the PRNG. We do not show the construction and the security theorem of this scheme since it is quite complex due to many cryptographic tools.

3.4.3 MuSig2

MuSig2 [NRS21] is a one-round multi-signature scheme with pre-processing. The multi-signature of it is the same form as the Schnorr signature. We show the construction of MuSig2 in Fig. 3.5.

In [NRS21], the authors prove the security of MuSig2 with and without the algebraic group model (AGM). Specifically, MuSig2 with $\nu = 4$ is proven secure under the AOMDL assumption without using AGM, and MuSig2 with $\nu = 2$ is proven secure under the AOMDL assumption in the AGM. We call the former and the latter MuSig2-1 and MuSig2-2, respectively. Below, we restate the security theorems of both of them.

Theorem 4 ([NRS21, Theorem 1])**.** *If there exists an adversary $\mathcal{A}$ that $(t_{\mathcal{A}}, Q_S, Q_H, N, \epsilon_{\mathcal{A}})$-breaks the (1-1)-MS-UF of* MuSig2-1*, then there exists an adversary $\mathcal{B}$ that $(t_{\mathcal{B}}, \epsilon_{\mathcal{B}})$-solves the $4Q_S$-AOMDL problem such that*

$$\epsilon_{\mathcal{B}} \geq \frac{\epsilon_{\mathcal{A}}^4}{Q_T^3} - \frac{32Q_T^2 + 22}{p}, \quad \text{and}$$
$$t_{\mathcal{B}} \leq 4t_{\mathcal{A}} + 4(N+6)Q_T t_{\mathrm{mul}} + O(NQ_T),$$

where $Q_T = 2Q_H + Q_S + 1$ and t_{mul} is the time of an scalar multiplication in $\mathbb{G}$.

$\mathsf{Setup}(1^\lambda)$:

$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$

Select $\mathsf{H_c}$ // $\mathsf{H_c} : \{0,1\}^* \to \mathbb{Z}_p$

Select $\mathsf{H_{non}}$ // $\mathsf{H_{ck}} : \{0,1\}^* \to \mathbb{Z}_p$

Select $\mathsf{H_{agg}}$ // $\mathsf{H_{agg}} : \{0,1\}^* \to \mathbb{Z}_p$

return $\mathsf{par} = (\mathbb{G}, p, G, \mathsf{H_c}, \mathsf{H_{ck}})$.

$\mathsf{Agg}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, (\mathsf{pm}_i, \mathsf{psig}_i)_{i \in [N]})$:

parse $(X_i)_{i \in [N]} \leftarrow \mathsf{vkList}$

parse $((R_{i,k})_{k \in [\nu]}, s_i)_{i \in [N]} \leftarrow (\mathsf{pm}_i, \mathsf{psig}_i)_{i \in [N]}$

for $i \in [N]$ **do** $t_i \leftarrow \mathsf{H_{agg}}(X_i, \mathsf{vkList})$

$\widetilde{\mathsf{pk}} \leftarrow \sum_{i=1}^{N} t_i X_i$

for $k \in [\nu]$ **do** $R_k \leftarrow \sum_{j=1}^{N} R_{j,k}$

$b \leftarrow \mathsf{H_{non}}(\widetilde{\mathsf{pk}}, (R_k)_{k \in [\nu]}, \mathsf{M})$

$\widetilde{R} \leftarrow \sum_{k=1}^{\nu} b^{k-1} R_k$

$\tilde{s} \leftarrow \sum_{i=1}^{N} s_i$

$\widetilde{\mathsf{sig}} \leftarrow (\widetilde{R}, \tilde{s})$

return $\widetilde{\mathsf{sig}}$

$\mathsf{Verify}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, \widetilde{\mathsf{sig}})$:

parse $(X_i)_{i \in [N]} \leftarrow \mathsf{vkList}$

parse $(\widetilde{R}, \tilde{s}) \leftarrow \widetilde{\mathsf{sig}}$

for $i \in [N]$ **do** $t_i \leftarrow \mathsf{H_{agg}}(X_i, \mathsf{vkList})$

$\widetilde{\mathsf{pk}} \leftarrow \sum_{i=1}^{N} t_i X_i$

$c \leftarrow \mathsf{H_c}(\widetilde{R}, \mathsf{M}, \widetilde{\mathsf{pk}})$

if $\left[\!\left[\widetilde{R} = \tilde{s}G - c \cdot \widetilde{\mathsf{pk}}\right]\!\right]$ **then**

  return 1

return 0

$\mathsf{KeyGen}(\mathsf{par}) \to (\mathsf{pk}, \mathsf{sk})$:

$x \xleftarrow{\$} \mathbb{Z}_p$

$X \leftarrow xG,$

$\mathsf{pk} \leftarrow X$

$\mathsf{sk} \leftarrow x$

return $(\mathsf{pk}, \mathsf{sk})$

$\mathsf{Sign}_0^{(1\text{-}1)}(\mathsf{par}, i, \mathsf{sk}_i)$:

for $k \in [\nu]$ **do**

  $r_{i,k} \xleftarrow{\$} \mathbb{Z}_p$

  $R_{i,k} \leftarrow r_{i,k} G$

$\mathsf{pm}_i \leftarrow (R_{i,k})_{k \in [\nu]}$

$\mathsf{st}_i \leftarrow (r_{i,k}, R_{i,k})_{k \in [\nu]}$

return $(\mathsf{pm}_i, \mathsf{st}_i)$

$\mathsf{Sign}_1^{(1\text{-}1)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_j)_{j \in [N] \setminus \{i\}})$:

parse $(X_j)_{j \in [N]} \leftarrow \mathsf{vkList}$

parse $(R_{j,k})_{j \in [N], k \in [\nu]} \leftarrow (\mathsf{pm}_j)_{j \in [N] \setminus \{i\}}$

parse $(r_{i,k}, R_{i,k})_{k \in [\nu]} \leftarrow \mathsf{st}_i$

for $j \in [N]$ **do**

  $t_j \leftarrow \mathsf{H_{agg}}(X_j, \mathsf{vkList})$

$\widetilde{\mathsf{pk}} \leftarrow \sum_{j=1}^{N} t_j X_j$

for $k \in [\nu]$ **do**

  $R_k \leftarrow \sum_{j=1}^{N} R_{j,k}$

$b \leftarrow \mathsf{H_{non}}(\widetilde{\mathsf{pk}}, (R_k)_{k \in [\nu]}, \mathsf{M})$

$\widetilde{R} \leftarrow \sum_{k=1}^{\nu} b^{k-1} R_k$

$c \leftarrow \mathsf{H_c}(\widetilde{R}, \mathsf{M}, \widetilde{\mathsf{pk}})$

$s_i \leftarrow x_i t_i c + \sum_{k=1}^{\nu} b^{k-1} r_i \mod p$

$\mathsf{psig}_i \leftarrow s_i$

return psig_i

Figure 3.5: The construction of MuSig2. $\mathsf{H_c}, \mathsf{H_{non}}$, and $\mathsf{H_{agg}}$ are modeled as random oracles. N is the number of the signers in vkList.

Theorem 5 ([NRS21, Theorem 2]). *If there exists an algebraic adversary $\mathcal{A}$ that $(t_{\mathcal{A}}, Q_S, Q_H, N, \epsilon_{\mathcal{A}})$-breaks the (1-1)-MS-UF of* MuSig2-2*, then there exists an adversary $\mathcal{B}$ that $(t_{\mathcal{B}}, \epsilon_{\mathcal{B}})$-solves the $2Q_S$-AOMDL problem such that*

$$\epsilon_{\mathcal{B}} \geq \epsilon_{\mathcal{A}} - \frac{26Q_T^3}{p}, \quad \text{and}$$
$$t_{\mathcal{B}} \leq t_{\mathcal{A}} + O(NQ_T)t_{\mathrm{mul}} + O(Q_T^3),$$

where $Q_T = 2Q_H + (N+2)(Q_S+1)$ and t_{mul} is the time of an scalar multiplication in $\mathbb{G}$.

DWMS [AB21] is a one-round multi-signature scheme with pre-processing that is similar to MuSig2. This scheme was proposed independently at the same time as MuSig2 was proposed. Since the construction of this scheme is similar to MuSig2, we do not show it. This scheme is proven secure under the OMDL problem in the AGM by using newly introduced the entwined sum problem. Below, we restate the security theorem of this scheme. Note that the running time overhead of the reduction is not evaluated in the security proof in [AB21]. We obtain the relationship of the running time in the following theorem by following the same argument as [BD21, Appendix A].

Theorem 6 ([AB21]). *If there exists an algebraic adversary $\mathcal{A}$ that $(t_{\mathcal{A}}, Q_S, Q_H, N, \epsilon_{\mathcal{A}})$-breaks the (1-1)-MS-UF of* DWMS*, then there exists an adversary $\mathcal{B}$ that $(t_{\mathcal{B}}, \epsilon_{\mathcal{B}})$-solves the $2Q_S$-AOMDL problem and an adversary $\mathcal{B}'$ that $(t_{\mathcal{B}'}, \epsilon_{\mathcal{B}'})$-solves the DL problem such that*

$$\epsilon_{\mathcal{B}} + \epsilon_{\mathcal{B}'} \geq \epsilon_{\mathcal{A}} - \frac{Q_S Q_H + 2Q_S + 4Q_H}{p} - \frac{Q_H}{\sqrt{p}} - \frac{Q_S}{p^2}, \quad \text{and}$$
$$t_{\mathcal{B}},\ t_{\mathcal{B}'} = O(t_{\mathcal{A}}).$$

Tessaro and Zhu proposed a one-round multi-signature scheme TZ with pre-processing [TZ23] based on the above schemes. Roughly, this scheme is a variant of MuSig2 based on the Okamoto signature scheme. This scheme is proven secure under the DL assumption. We omit the construction of this scheme. Below, we restate the security theorem of this scheme.

Theorem 7 ([TZ23]). *If there exists an adversary $\mathcal{A}$ that $(t_{\mathcal{A}}, Q_S, Q_H, N, \epsilon_{\mathcal{A}})$-breaks the (1-1)-MS-UF of* TZ*, then there exists an adversary $\mathcal{B}$ that $(t_{\mathcal{B}}, \epsilon_{\mathcal{B}})$-solves the DL problem such that*

$$\epsilon_{\mathcal{B}} \geq \frac{\epsilon_{\mathcal{A}}^4}{2Q_T^3} - \frac{8Q_T + 8}{p}, \quad \text{and}$$
$$t_{\mathcal{B}} \approx 4t_{\mathcal{A}},$$

where $Q_T = Q_H + Q_S + 1$.

3.4.4 HBMS

Bellare and Dai proposed an improvement of mBCJ as HBMS [BD21]. We show the construction of HBMS in Fig. 3.6.

Below, we restate the security theorems of HBMS.

Theorem 8 ([BD21]). *If there exists an adversary $\mathcal{A}$ that $(t_{\mathcal{A}}, Q_S, Q_H, N, \epsilon_{\mathcal{A}})$-breaks the 2-MS-UF of* HBMS, *then there exists an adversary $\mathcal{B}$ that $(t_{\mathcal{B}}, \epsilon_{\mathcal{B}})$-solves the DL problem such that*

$$\epsilon_{\mathcal{B}} \geq \frac{\epsilon^4}{Q_H^3 e^4 (Q_S+1)^4} - \frac{3}{p}, \quad \text{and}$$
$$t_{\mathcal{B}} \approx 4t_{\mathcal{A}},$$

where e is the base of the natural logarithm.

They also proved that the scheme is tightly secure under the DL assumption in the AGM.

Theorem 9 ([BD21]). *If there exists an algebraic adversary $\mathcal{A}$ that $(t_{\mathcal{A}}, Q_S, Q_H, N, \epsilon_{\mathcal{A}})$-breaks the 2-MS-UF of* HBMS, *then there exists an adversary $\mathcal{B}$ that $(t_{\mathcal{B}}, \epsilon_{\mathcal{B}})$-solves the DL problem such that*

$$\epsilon_{\mathcal{B}} \geq \epsilon_{\mathcal{A}} - \frac{Q_H(Q_H+1)}{p}, \quad \text{and}$$
$$t_{\mathcal{B}} \approx t_{\mathcal{A}}.$$

We call HBMS as HBMS-AGM when the security of HBMS is given by the following theorem.

Lee and Kim proposed a two-round scheme LK [LK22] based on HBMS and the Okamoto signature scheme. They proved that this scheme is secure under the DL assumption in the AGM. Note that their unforgeability game is slightly different from one in Fig. 2.7. Specifically, as the winning condition, the challenger checks $\mathsf{M}^* \in \mathsf{Q_M}$ instead of $(\mathsf{M}^*, \mathsf{vkList}^*) \in \mathsf{Q_M}$. We omit the construction of this scheme and the security theorem.

3.4.5 Pan-Wagner Schemes

Pan and Wagner proposed two two-round multi-signature schemes based on the DDH assumption [PW23]. This work is concurrent and independent work of our work. These schemes are constructed by combining the technique of mBCJ and the Katz-Wang signature schemes. The first scheme PW-1 achieves tight security but key aggregation is not supported. The second scheme PW-2

$\mathsf{Setup}(1^\lambda)$:

$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$
Select $\mathsf{H_c}$ // $\mathsf{H_c} : \{0,1\}^* \to \mathbb{Z}_p$
Select $\mathsf{H_{ck}}$ // $\mathsf{H_{ck}} : \{0,1\}^* \to \mathbb{G}$
Select $\mathsf{H_{agg}}$ // $\mathsf{H_{agg}} : \{0,1\}^* \to \mathbb{Z}_p$
return $\mathsf{par} = (\mathbb{G}, p, G, \mathsf{H_c}, \mathsf{H_{ck}}, \mathsf{H_{agg}})$.

$\mathsf{KeyGen}(\mathsf{par}) \to (\mathsf{pk}, \mathsf{sk})$:

$x \xleftarrow{\$} \mathbb{Z}_p$
$X \leftarrow xG$
$\mathsf{pk} \leftarrow X$
$\mathsf{sk} \leftarrow x$
return $(\mathsf{pk}, \mathsf{sk})$

$\mathsf{Agg}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, (\mathsf{pm}_i, \mathsf{psig}_i)_{i\in[N]})$:

parse $(X_i)_{i\in[N]} \leftarrow \mathsf{vkList}$
parse $(T_i, d_i, s_i)_{i\in[N]} \leftarrow (\mathsf{pm}_i, \mathsf{psig}_i)_{i\in[N]}$
$\widetilde{T} \leftarrow \sum_{i=1}^{N} T_i$
$\tilde{d} \leftarrow \sum_{i=1}^{N} d_i \mod p$
$\tilde{s} \leftarrow \sum_{i=1}^{N} s_i \mod p$
$\widetilde{\mathsf{sig}} \leftarrow (\widetilde{T}, \tilde{d}, \tilde{s})$
return $\widetilde{\mathsf{sig}}$

$\mathsf{Sign}_1^{(2)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i)$:

parse $(X_j)_{j\in[N]} \leftarrow \mathsf{vkList}$
for $j \in [N]$ **do**
  $t_j \leftarrow \mathsf{H_{agg}}(X_j, \mathsf{vkList})$
$\widetilde{\mathsf{pk}} \leftarrow \sum_{j=1}^{N} t_j X_j$
$U \leftarrow \mathsf{H_{ck}}(\mathsf{M}, \mathsf{vkList})$
$r_i, d_i \xleftarrow{\$} \mathbb{Z}_p$
$T_i \leftarrow d_i U + r_i G$
$\mathsf{pm}_i \leftarrow T_i$
$\mathsf{st}_i \leftarrow (r_i, d_i, t_i, T_i, \widetilde{\mathsf{pk}})$
return $(\mathsf{pm}_i, \mathsf{st}_i)$

$\mathsf{Verify}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, \widetilde{\mathsf{sig}})$:

parse $(X_i)_{i\in[N]} \leftarrow \mathsf{vkList}$
parse $(\widetilde{T}, \tilde{d}, \tilde{s}) \leftarrow \widetilde{\mathsf{sig}}$
for $i \in [N]$ **do**
  $t_i \leftarrow \mathsf{H_{agg}}(X_i, \mathsf{vkList})$
$\widetilde{\mathsf{pk}} \leftarrow \sum_{i=1}^{N} t_i X_i$
$c \leftarrow \mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})$
$U \leftarrow \mathsf{H_{ck}}(\mathsf{M}, \mathsf{vkList})$
if $[\![\widetilde{T} = \tilde{d}U + \tilde{s}G - c \cdot \widetilde{\mathsf{pk}}]\!]$ **then**
  return 1
return 0

$\mathsf{Sign}_2^{(2)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_j)_{j\in[N]\setminus\{i\}})$:

parse $(T_j)_{j\in[N]\setminus\{i\}} \leftarrow (\mathsf{pm}_j)_{j\in[N]\setminus\{i\}}$
parse $(r_i, d_i, t_i, T_i, \widetilde{\mathsf{pk}}) \leftarrow \mathsf{st}_i$
$\widetilde{T} \leftarrow \sum_{j=1}^{N} T_j$
$c \leftarrow \mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})$
$s_i \leftarrow x_i t_i c + r_i \mod p$
$\mathsf{psig}_i \leftarrow (d_i, s_i)$
return psig_i

Figure 3.6: The construction of HBMS. $\mathsf{H_c}, \mathsf{H_{ck}}$, and $\mathsf{H_{agg}}$ are modeled as random oracles. N is the number of the signers in vkList.

supports key aggregation and achieves a small reduction loss. We omit the construction of them.

Below, we restate the security theorems of them.

Theorem 10 ([PW23]). *If there exists an adversary $\mathcal{A}$ that $(t_{\mathcal{A}}, Q_S, Q_H, N, \epsilon_{\mathcal{A}})$-breaks the 2-MS-UF of* PW-1*, then there exists an adversary $\mathcal{B}$ that $(t_{\mathcal{B}}, \epsilon_{\mathcal{B}})$-solves the DDH problem such that*

$$\epsilon_{\mathcal{B}} \geq \frac{\epsilon_{\mathcal{A}}}{4} - \frac{Q_H^2 + 2Q_H + 3Q_S + 5}{p} - \frac{Q_H}{2^{\lambda-2}}, \quad \text{and}$$
$$t_{\mathcal{B}} \approx t_{\mathcal{A}}.$$

Theorem 11 ([PW23]). *If there exists an adversary $\mathcal{A}$ that $(t_{\mathcal{A}}, Q_S, Q_H, N, \epsilon_{\mathcal{A}})$-breaks the 2-MS-UF of* PW-2*, then there exists an adversary $\mathcal{B}$ that $(t_{\mathcal{B}}, \epsilon_{\mathcal{B}})$-solves the DDH problem such that*

$$\epsilon_{\mathcal{B}} \geq \frac{\epsilon_{\mathcal{A}}}{8Q_S} - \frac{3Q_H^2 + 3Q_S + 4}{p}, \quad \text{and}$$
$$t_{\mathcal{B}} \approx t_{\mathcal{A}}.$$

Note that, in their formal theorem, these schemes are proven secure under not only the DDH assumption but also the variant of the DDH assumption. Since the variant is tightly equivalent to the DDH assumption, which is explained in [PW23], we obtain the relationship of the advantages by regarding the variant as the DDH assumption.

Remark 2. *Our scheme which will be proposed in the next chapter achieves a more efficient signature size and communication complexity than* PW-2 *although ours and* PW-2 *have something in common. Specifically, as shown in Table 5.1, they have common points, e.g., a security assumption, a reduction loss, and a standardized EC for 128-bit security. While the signature size of* PW-2 *is $5|\mathbb{Z}_p|$, that of ours achieves $3|\mathbb{Z}_p|$ where $|\mathbb{Z}_p|$ is the size of $\mathbb{Z}_p$.*

Whereas Pan and Wagner dedicated the effort to present their construction in a generic and modular way, we trade generality and modularity for more efficiency. Our improvement is a benefit of this. PW-2 *is an instantiation of their generic construction.* PW-2 *is constructed by combining a special commitment scheme and other building blocks in a generic manner. The scheme requires the binding property of the commitment scheme to prove the unforgeability. On the other hand, we construct our scheme in a specific way to be able to directly prove the unforgeability without using the binding property of the commitment scheme. In short, in our scheme, the binding property is not a necessary condition. Then, we can reduce the size of the commitment key, the commitment, and the decommitment. This gives a smaller signature size and communication complexity. However, our scheme cannot be captured by Pan-Wagner's generic construction because the commitment scheme used in ours deviates from the syntax of the special commitment scheme.*

3.5 Proof of Theorem 3

Here, we prove Theorem 3, establishing the security of $\mathsf{mBCJ\text{-}PPK}$ under the DL assumption.

Proof of Theorem 3. We construct $\mathcal{B}$ which solves the DL problem using $\mathcal{A}$. $\mathcal{B}$ on input $(\mathbb{G}, p, G)$ and X, which are a parameter and an instance of the DL problem, outputs x such that $X = xG$. Specifically, it runs the forking algorithm in Lemma 1 for an algorithm $\mathcal{C}$ that internally runs $\mathcal{A}$ to extract x.

To construct $\mathcal{B}$, we construct another algorithm $\mathcal{C}$ as follows. On input $(\mathbb{G}, p, G, X)$, a random tape ρ and $h_1, \ldots, h_{N(Q_H+Q_S+1)} \in \mathbb{Z}_p$, it internally runs $\mathcal{A}$ on input $(\mathbb{G}, p, G)$ and X as a public parameter par and a public key pk. It initiates a counter $\mathsf{ctr} = 1$, tables $\mathsf{T}_{\mathsf{H_{ck}}}[\cdot]$, $\mathsf{T}_{\mathsf{H_c}}[\cdot]$, $\mathsf{T}_{\mathrm{td}}[\cdot]$, $\mathsf{Q}_{\mathsf{st}}[\cdot]$ to $\perp$, and a set Q_{M} to $\emptyset$, where $\mathsf{T}_{\mathsf{H_{ck}}}[\cdot]$ and $\mathsf{T}_{\mathsf{H_c}}[\cdot]$ are random oracle tables for $\mathsf{H_{ck}}$ and $\mathsf{H_c}$, respectively, and $\mathsf{T}_{\mathrm{td}}[\cdot]$ is a table to store the trapdoors of the commitment keys. Also, it responds to random oracle queries and signing queries as follows.

Random Oracle $\mathsf{H_{ck}}(\mathsf{M}, \mathsf{vkList})$: It returns $\mathsf{T}_{\mathsf{H_{ck}}}[\mathsf{M}, \mathsf{vkList}]$ if $\mathsf{T}_{\mathsf{H_{ck}}}[\mathsf{M}, \mathsf{vkList}]$ is already defined. If $\mathsf{T}_{\mathsf{H_{ck}}}[\mathsf{M}, \mathsf{vkList}]$ is undefined, it responds as follows. It chooses a bit b which becomes 1 with probability $\delta = Q_S/(Q_S + 1)$. If $b = 1$, it chooses $(\omega_{1,1}, \omega_{1,2}, \omega_{1,3}) \xleftarrow{\$} \mathbb{Z}_p^3$, computes $U_1 \leftarrow \omega_{1,1}G$, $U_2 \leftarrow \omega_{1,2}G$, and $U_3 \leftarrow \omega_{1,3}X$. If $b = 0$, it chooses $(\omega_{0,1}, \omega_{0,2}, \omega_{0,3}) \xleftarrow{\$} \mathbb{Z}_p^3$, computes $U_1 \leftarrow \omega_{0,1}G$, $U_2 \leftarrow \omega_{0,2}X$, and $U_3 \leftarrow \omega_{0,3}G$. It assigns $\mathsf{T}_{\mathsf{H_{ck}}}[\mathsf{M}, \mathsf{vkList}] \leftarrow (U_1, U_2, U_3)$ and $\mathsf{T}_{\mathrm{td}}[\mathsf{M}, \mathsf{vkList}] \leftarrow (b, \omega_{b,1}, \omega_{b,2}, \omega_{b,3})$ and returns $\mathsf{T}_{\mathsf{H_{ck}}}[\mathsf{M}, \mathsf{vkList}]$.

Random Oracle $\mathsf{H_c}(X_i, \widetilde{T}_1, \widetilde{T}_2, \mathsf{vkList}, \mathsf{M})$: If $\mathsf{T}_{\mathsf{H_{ck}}}[\mathsf{M}]$ is undefined, it makes a query $\mathsf{H_{ck}}(\mathsf{M})$. If $\mathsf{T}_{\mathsf{H_c}}[X_i, \widetilde{T}_1, \widetilde{T}_2, \mathsf{vkList}, \mathsf{M}]$ is already defined, it returns h where $(h, J) = \mathsf{T}_{\mathsf{H_c}}[X_i, \widetilde{T}_1, \widetilde{T}_2, \mathsf{vkList}, \mathsf{M}]$. If $\mathsf{T}_{\mathsf{H_c}}[X_i, \widetilde{T}_1, \widetilde{T}_2, \mathsf{vkList}, \mathsf{M}]$ is undefined, it responds as follows.

Case $(X \in \mathsf{vkList})$**:** For j s.t. $X \neq X_j \in \mathsf{vkList}$, it assigns $\mathsf{T}_{\mathsf{H_c}}[X_j, \widetilde{T}_1, \widetilde{T}_2, \mathsf{vkList}, \mathsf{M}] \leftarrow (h_{\mathsf{ctr}}, \mathsf{ctr})$ and sets $\mathsf{ctr} \leftarrow \mathsf{ctr} + 1$. After that, for j s.t. $X = X_j \in \mathsf{vkList}$, it assigns $\mathsf{T}_{\mathsf{H_c}}[X_i, \widetilde{T}_1, \widetilde{T}_2, \mathsf{vkList}, \mathsf{M}] \leftarrow (h_{\mathsf{ctr}}, \mathsf{ctr})$ and sets $\mathsf{ctr} \leftarrow \mathsf{ctr} + 1$. It returns h where $(h, J) = \mathsf{T}_{\mathsf{H_c}}[X_i, \widetilde{T}_1, \widetilde{T}_2, \mathsf{vkList}, \mathsf{M}]$.

Case $(X \notin \mathsf{vkList})$**:** It assigns $\mathsf{T}_{\mathsf{H_c}}[X_i, \widetilde{T}_1, \widetilde{T}_2, \mathsf{vkList}, \mathsf{M}] \leftarrow (h_{\mathsf{ctr}}, \mathsf{ctr})$, sets $\mathsf{ctr} \leftarrow \mathsf{ctr}+1$, and returns h where $(h, J) = \mathsf{T}_{\mathsf{H_c}}[X_i, \widetilde{T}_1, \widetilde{T}_2, \mathsf{vkList}, \mathsf{M}]$.

Signing Queries: Note that the honest signer is corresponding to pk_1.

$\mathcal{O}_{\mathsf{Sign}_1^{(2)}}(\mathsf{sid}, \mathsf{vkList}, \mathsf{M})$: If $(\mathsf{pk} \in \mathsf{vkList}) \wedge (\mathsf{Q}_{\mathsf{st}}[\mathsf{sid}, 1] = \bot) \wedge (|\mathsf{vkList}| \leq N)$ is not true, it returns $\bot$. It sets $\mathsf{Q}_{\mathsf{M}} \leftarrow \mathsf{Q}_{\mathsf{M}} \cup \{(\mathsf{vkList}, \mathsf{M})\}$. It makes a query $\mathsf{H}_{\mathsf{ck}}(\mathsf{M}, \mathsf{vkList})$ if $\mathsf{T}_{\mathsf{H}_{\mathsf{ck}}}[\mathsf{M}, \mathsf{vkList}]$ is undefined. It looks up $(b, \omega_{b,1}, \omega_{b,2}, \omega_{b,3})$ from $\mathsf{T}_{\mathrm{td}}[\mathsf{M}, \mathsf{vkList}]$. If $b = 0$ holds, then it halts with output $(0, \emptyset)$. Otherwise, it chooses $(u, v, w) \xleftarrow{\$} \mathbb{Z}_p^3$, computes $T_{1,1} \leftarrow uG$ and $T_{1,2} \leftarrow vG - wX$, stores $\mathsf{Q}_{\mathsf{st}}[\mathsf{sid}, 1] \leftarrow (\mathsf{vkList}, \mathsf{M}, (u, v, w, T_{1,1}, T_{1,2}))$, and sets $\mathsf{pm}_1 \leftarrow (T_{1,1}, T_{1,2})$. It returns pm_1.

$\mathcal{O}_{\mathsf{Sign}_2^{(2)}}(\mathsf{sid}, (\mathsf{pm}_j)_{j \in [|\mathsf{vkList}|] \setminus \{1\}})$: If $(\mathsf{Q}_{\mathsf{st}}[\mathsf{sid}, 1] \neq \bot) \wedge (\mathsf{Q}_{\mathsf{st}}[\mathsf{sid}, 2] = \bot)$ is not true, it returns $\bot$. It looks up $(\mathsf{vkList}, \mathsf{M}, (u, v, w, T_{1,1}, T_{1,2})) \leftarrow \mathsf{Q}_{\mathsf{st}}[\mathsf{sid}, 1]$ and $(b, \omega_{b,1}, \omega_{b,2}, \omega_{b,3})$ from $\mathsf{T}_{\mathrm{td}}[\mathsf{M}, \mathsf{vkList}]$. If $\omega_{1,3} = 0$, it halts with output $(0, \emptyset)$. Otherwise, it sets $n \leftarrow |\mathsf{vkList}|$ and computes $\widetilde{T}_1 \leftarrow \sum_{j=1}^n T_{j,1}$, $\widetilde{T}_2 \leftarrow \sum_{j=1}^n T_{j,2}$, $c \leftarrow \mathsf{H}_{\mathsf{c}}(X, \widetilde{T}_1, \widetilde{T}_2, \mathsf{vkList}, \mathsf{M})$, $\beta_1 \leftarrow (c - w)/\omega_{1,3} \mod p$, $\alpha_1 \leftarrow u - \omega_{1,2}\beta_1 \mod p$, and $s_1 \leftarrow v - \omega_{1,1}\alpha_1 \mod p$. It sets $\mathsf{psig}_1 \leftarrow (s_1, \alpha_1, \beta_1)$ and $\mathsf{Q}_{\mathsf{st}}[\mathsf{sid}, 2] \leftarrow \mathsf{psig}_1$. It returns psig_1.

If $\mathcal{A}$'s forgery $(\mathsf{M}^*, \mathsf{vkList}^*, \widetilde{\mathsf{sig}}^* = (\widetilde{R}^*, \tilde{s}^*, \tilde{\alpha}^*, \tilde{\beta}^*))$ does not satisfy $(\mathsf{pk} \in \mathsf{vkList}^*) \wedge (|\mathsf{vkList}^*| \leq N) \wedge ((\mathsf{vkList}^*, \mathsf{M}^*) \notin \mathsf{Q}_{\mathsf{M}}) \wedge (\mathsf{Verify}(\mathsf{par}, \mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*) = 1)$ or $b = 0$ where $(b, \omega_{b,1}, \omega_{b,2}, \omega_{b,3}) = \mathsf{T}_{\mathrm{td}}[\mathsf{M}^*, \mathsf{vkList}^*]$, then $\mathcal{C}$ halts with output $(0, \emptyset)$. Otherwise, $\mathcal{C}$ can get a forgery $(\mathsf{M}^*, \mathsf{vkList}^*, \widetilde{\mathsf{sig}}^* = (\widetilde{R}_1^*, \tilde{s}^*, \tilde{\alpha}^*, \tilde{\beta}^*))$ s.t. $X \in \mathsf{vkList}^*$, $|\mathsf{vkList}^*| \leq N$, $(\mathsf{vkList}^*, \mathsf{M}^*) \notin \mathsf{Q}_{\mathsf{M}}$, $\mathsf{Verify}(\mathsf{par}, \mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*) = 1$, and $(0, \omega_{0,1}, \omega_{0,2}, \omega_{0,3}) = \mathsf{T}_{\mathrm{td}}[\mathsf{M}^*, \mathsf{vkList}^*]$. Let J be the integer such that $(c, J) = \mathsf{T}_{\mathsf{H}_{\mathsf{c}}}[X, \widetilde{T}_1^*, \widetilde{T}_2^*, \mathsf{vkList}^*, \mathsf{M}^*]$. It outputs $(J, \sigma = (\mathsf{vkList}^*, \omega_{0,1}, \omega_{0,2}, \omega_{0,3}, \tilde{s}^*, \tilde{\alpha}^*, \tilde{\beta}^*, (c_i)_{i=1}^{|\mathsf{vkList}^*|}))$ where $(c_i, I_i) = \mathsf{T}_{\mathsf{H}_{\mathsf{c}}}[X_i, \widetilde{T}_1^*, \widetilde{T}_2^*, \mathsf{vkList}^*, \mathsf{M}^*]$.

Following the same argument in the proof of [DEF+19, Theorem 3], the distribution of $\mathcal{C}$'s responses of the signing oracles is identical to the distribution of the honest signer's responses.

Let acc be the probability that $\mathcal{C}$ outputs $J > 0$. Also, we define the following events.

- E_1: $\mathcal{A}$'s forgery satisfies the winning conditions in the game of the security definition.
- E_2: $\mathcal{A}$'s forgery satisfies $b = 1$ where $(b, \omega_{b,1}, \omega_{b,2}, \omega_{b,3}) = \mathsf{T}_{\mathrm{td}}[\mathsf{M}^*, \mathsf{vkList}^*]$.
- E_3: $\mathcal{C}$ halts because of $b = 0$ in $\mathcal{O}_{\mathsf{Sign}_1^{(2)}}$.
- E_4: $\mathcal{C}$ halts because of $\omega_{1,3} = 0$ in $\mathcal{O}_{\mathsf{Sign}_2^{(2)}}$.

Then, we obtain the following equations.

$$\begin{aligned}\mathsf{acc} &= \Pr[\mathsf{E}_1 \wedge \mathsf{E}_2 \wedge \overline{\mathsf{E}_3} \wedge \overline{\mathsf{E}_4}] \\ &= \Pr[\overline{\mathsf{E}_3}]\Pr[\overline{\mathsf{E}_4}|\overline{\mathsf{E}_3}]\Pr[\mathsf{E}_1|\overline{\mathsf{E}_3} \wedge \overline{\mathsf{E}_4}]\Pr[\mathsf{E}_2|\mathsf{E}_1 \wedge \overline{\mathsf{E}_3} \wedge \overline{\mathsf{E}_4}]\end{aligned}$$

We first evaluate $\Pr[\overline{\mathsf{E}_3}]$. $\overline{\mathsf{E}_3}$ means that, for all messages M chosen by $\mathcal{A}$ as signing queries, the bit b becomes 1 when $\mathsf{T}_{\mathsf{H}_{\mathsf{ck}}}[\mathsf{M}, \mathsf{vkList}]$ is defined. Since the distributions of the responses of $\mathsf{T}_{\mathsf{H}_{\mathsf{ck}}}[\mathsf{M}, \mathsf{vkList}]$ are identical in both cases $b = 0$ and $b = 1$, $\mathcal{A}$ does not obtain information on $\mathsf{T}_{\mathsf{H}_{\mathsf{ck}}}[\mathsf{M}, \mathsf{vkList}]$ from them. Also, it only now $\mathsf{T}_{\mathsf{H}_{\mathsf{ck}}}[\mathsf{M}, \mathsf{vkList}] = 1$ for $(\mathsf{M}, \mathsf{vkList})$ queried to signing oracles. Thus, the view of $\mathcal{A}$ does not leak $\mathsf{T}_{\mathsf{H}_{\mathsf{ck}}}[\mathsf{M}, \mathsf{vkList}] = 0$. Because $b = 1$ occurs with probability δ and $\mathcal{A}$ makes at most Q_S signing queries, we obtain $\Pr[\overline{\mathsf{E}_3}] \geq \delta^{Q_S}$.

Next, We evaluate $\Pr[\overline{\mathsf{E}_4}|\overline{\mathsf{E}_3}]$. Conditioned on $\overline{\mathsf{E}_3}$, $\omega_{1,3}$ is picked uniformly at random from $\mathbb{Z}_p$. $\Pr[\overline{\mathsf{E}_4}|\overline{\mathsf{E}_3}]$ is equal to the probability that $\omega_{1,3} \neq 0$ for all messages M queried to the signing oracle. Thun, we get $\Pr[\overline{\mathsf{E}_4}|\overline{\mathsf{E}_3}] = (1 - 1/p)^{Q_S}$.

Now we evaluate $\Pr[\mathsf{E}_1|\overline{\mathsf{E}_3} \wedge \overline{\mathsf{E}_4}]$. Conditioned on $\overline{\mathsf{E}_3} \wedge \overline{\mathsf{E}_4}$, $\mathcal{C}$ does not halt the game. Also, the condition of E_1 is identical to that in the unforgeability game. Thus, we obtain $\Pr[\mathsf{E}_1|\overline{\mathsf{E}_3} \wedge \overline{\mathsf{E}_4}] = \epsilon_{\mathcal{A}}$.

Finally, we evaluate $\Pr[\mathsf{E}_2|\mathsf{E}_1 \wedge \overline{\mathsf{E}_3} \wedge \overline{\mathsf{E}_4}]$. Because the distribution of $\mathsf{H}_{\mathsf{ck}}(\cdot)$ is independent of the value of bits, $\mathcal{A}$ cannot know the value of the bit for $(\mathsf{M}^*, \mathsf{vkList}^*)$. Thus, the event that $b = 0$ for M^* happens with probability $1 - \delta$. Therefore, $\Pr[\mathsf{E}_2|\mathsf{E}_1 \wedge \overline{\mathsf{E}_3} \wedge \overline{\mathsf{E}_4}] = 1 - \delta$.

From the above arguments, we obtain $\mathsf{acc} \geq \delta^{Q_S} (1 - 1/q)^{Q_S} \epsilon_{\mathcal{A}}(1 - \delta)$. Since we set $\delta = Q_S/(Q_S + 1)$, we have

$$\begin{aligned}\mathsf{acc} &= \frac{1}{(1 + 1/Q_S)^{Q_S}} \left(1 - \frac{1}{p}\right)^{Q_S} \epsilon_{\mathcal{A}} \frac{1}{Q_S + 1} \\ &\geq \left(1 - \frac{Q_S}{p}\right) \frac{\epsilon_{\mathcal{A}}}{e(Q_S + 1)}\end{aligned}$$

by using the facts that $(1 + 1/Q_S)^{Q_S} < e$ for $Q_S \geq 0$, where e is the base of the natural logarithm, and $(1 + a)^b \geq 1 + ab$ for $a \geq -1$ and a natural number b.

Let $t_{\mathcal{C}}$ be the running time of $\mathcal{C}$. We assume that t_{mul} time is required for one scalar multiplication in $\mathbb{G}$, and unit time is required for the other non-cryptographic operations. $\mathcal{C}$ runs $\mathcal{A}$ at once. For time to answer random oracle queries, we consider only the case of H_{c} because H_{c} takes a longer time than H_{ck}. $\mathcal{C}$ makes one query to H_{ck} and executes $O(N)$ other non-cryptographic operations to respond to a query to H_{c}. Three scalar multiplications and $O(1)$ other non-cryptographic operations are required for one random oracle query to H_{ck}. Thus, in total, $\mathcal{C}$ executes three scalar multiplications and $O(N)$ other non-cryptographic operations to respond to one random oracle query to H_{c}. In each signing query, $\mathcal{C}$ needs to execute one random oracle query to H_{ck}, three scalar multiplications, and $O(N)$ other non-cryptographic operations. The verification involves at most $(N+5)$ scalar multiplications, one random oracle query to H_{ck}, and $O(N)$ other non-cryptographic operations. Also $\mathcal{C}$ needs $O(N)$ other non-cryptographic operations to output (J, σ) after checking $\mathcal{A}$'s output. Therefore, $t_{\mathcal{C}}$ is at most $t_{\mathcal{A}} + (3Q_H + 6Q_S + N + 8)t_{\mathrm{mul}} + O(N(Q_S + Q_H))$.

$\mathcal{B}$ runs the forking algorithm $\mathsf{Fork}_{\mathcal{C}}(\mathbb{G}, p, G, X)$ in Lemma 1. If $\mathsf{Fork}_{\mathcal{C}}$ succeeds in outputting $(1, \sigma, \sigma')$, $\mathcal{B}$ obtains $\sigma = (\mathsf{vkList}^*, \omega_{0,1}, \omega_{0,2}, \omega_{0,3}, \tilde{s}^*, \tilde{\alpha}^*, \tilde{\beta}^*, (c_i)_{i=1}^{|\mathsf{vkList}^*|})$ and $\sigma' = (\mathsf{vkList}'^*, \omega'_{0,1}, \omega'_{0,2}, \omega'_{0,3}, \tilde{s}'^*, \tilde{\alpha}'^*, \tilde{\beta}'^*, (c'_i)_{i=1}^{|\mathsf{vkList}'^*|})$ such that $(\mathsf{vkList}^*, \omega_{0,1}, \omega_{0,2}, \omega_{0,3}, (c_i)_{i \in K^*}) = (\mathsf{vkList}'^*, \omega'_{0,1}, \omega'_{0,2}, \omega'_{0,3}, (c'_i)_{i \in K'^*})$, $X \in \mathsf{vkList}^*$, $(c_i = c_j) \wedge (c'_{i'} = c'_{j'}) \wedge (c_i \neq c_{i'})$ for all $i, j \in [|\mathsf{vkList}^*|] \setminus K^*$ and $i', j' \in [|\mathsf{vkList}'^*|] \setminus K'^*$,

$$\tilde{\alpha}^* G + \tilde{\beta}^*(\omega_{0,2} X) = \tilde{\alpha}'^* G + \tilde{\beta}'^*(\omega'_{0,2} X), \text{ and,}$$
$$\tilde{\alpha}^*(\omega_{0,1} G) + \tilde{\beta}^*(\omega_{0,3} G) + \tilde{s}^* G - \sum_{j=1}^{|\mathsf{vkList}^*|} c_j X_j$$
$$= \tilde{\alpha}'^*(\omega_{0,1} G) + \tilde{\beta}'^*(\omega'_{0,3} G) + \tilde{s}'^* G - \sum_{j=1}^{|\mathsf{vkList}'^*|} c'_j X'_j$$

where K^* and K'^* are the sets of the indices s.t. $X \neq X_i \in \mathsf{vkList}^*$ and $X \neq X_i \in \mathsf{vkList}'^*$, respectively. Due to the above conditions, $\mathcal{B}$ can obtain the following equations

$$(\tilde{\alpha}^* - \tilde{\alpha}'^*)G = (\tilde{\beta}'^* - \tilde{\beta}^*)\omega_{0,2} X, \text{ and}$$
$$((\tilde{\alpha}^* - \tilde{\alpha}'^*)\omega_{0,1} + (\tilde{\beta}^* - \tilde{\beta}'^*)\omega_{0,3} + (\tilde{s}^* - \tilde{s}'))G = -(|\mathsf{vkList}^*| - |K^*|)(c - c')X$$

where c and c' are c_i and c'_i for $i \in [|\mathsf{vkList}^*|] \setminus K^*$ and $i \in [|\mathsf{vkList}'^*|] \setminus K'^*$, respectively. $(|\mathsf{vkList}^*| - |K^*|) \neq 0$ holds because vkList^* includes at least one X. Therefore, $\mathcal{B}$ can compute and output the discrete logarithm x of X as follows.

Case $(\tilde{\beta}'^* \neq \tilde{\beta}^*) \wedge (\omega_{0,2} \neq 0)$**:** $\mathcal{B}$ outputs x as

$$\frac{\tilde{\alpha}^* - \tilde{\alpha}'^*}{(\tilde{\beta}'^* - \tilde{\beta}^*)\omega_{0,2}}.$$

Case $(\tilde{\beta}'^* \neq \tilde{\beta}^*) \wedge (\omega_{0,2} = 0)$**:** In this case, $\tilde{\alpha}^* = \tilde{\alpha}'^*$ holds. Thus, $\mathcal{B}$ outputs x as

$$-\frac{(\tilde{\beta}^* - \tilde{\beta}'^*)\omega_{0,3} + (\tilde{s}^* - \tilde{s}')}{(|\mathsf{vkList}^*| - |K^*|)(c - c')}.$$

Case $\tilde{\beta}'^* = \tilde{\beta}^*$**:** In this case, $\tilde{\alpha}^* = \tilde{\alpha}'^*$ holds. Thus, $\mathcal{B}$ outputs x as

$$-\frac{\tilde{s}^* - \tilde{s}'}{(|\mathsf{vkList}^*| - |K^*|)(c - c')}.$$

To complete this proof, we evaluate the success probability and the running time of $\mathcal{B}$. Because $\mathcal{B}$ can output the solution of the DL problem if $\mathsf{Fork}_{\mathcal{C}}$ outputs $(1, \sigma, \sigma')$, $\mathcal{B}$ can solve the DL problem with the probability that $\mathsf{Fork}_{\mathcal{C}}$ outputs $(1, \sigma, \sigma')$ in time as same as the running time of $\mathsf{Fork}_{\mathcal{C}}$. Applying the Lemma 1, $\mathcal{B}$ solves the DL problem with probability at least ϵ such that

$$\begin{aligned}
\epsilon &\geq \mathsf{acc}\left(\frac{\mathsf{acc}}{N(Q_H + Q_S + 1)} - \frac{1}{p}\right) \\
&\geq \frac{\mathsf{acc}^2}{N(Q_H + Q_S + 1)} - \frac{1}{p} \\
&\geq \left(1 - \frac{Q_S}{p}\right)^2 \frac{\epsilon_{\mathcal{A}}^2}{N(Q_H + Q_S + 1)e^2(Q_S + 1)^2} - \frac{1}{p}.
\end{aligned}$$

Because $\mathcal{B}$ runs $\mathcal{C}$ twice, the running time of $\mathcal{B}$ is at most twice as long as the running time of $\mathcal{C}$. Thus, the running of $\mathcal{B}$ is at most $2t_{\mathcal{A}} + (6Q_H + 12Q_S + 2N + 16)t_{\mathrm{mul}} + O(N(Q_S + Q_H))$. This completes the proof. □

Chapter 4

New Two-Round Multi-Signature Schemes with Small Reduction Loss

In this chapter, we propose a new two-round multi-signature scheme with a small reduction loss and an improved one. Specifically, we first propose a new two-round multi-signature scheme HBMSDDH-1 and show that it satisfies the slightly weak unforgeability, which is described in Section 2.4.4. After that, we propose a variant of HBMSDDH-1 achieving the slightly strong unforgeability, which we call it HBMSDDH-2 hereafter. The modification is very small and the construction of this scheme is almost the same as HBMSDDH-1.

Toward constructing a two-round scheme with a small reduction loss, we attempt to combine the Katz-Wang DDH-based signature scheme [GJKW07] and the DL-based two-round multi-signature scheme HBMS [BD21]. The Katz-Wang DDH-based signature scheme achieves tight security under the DDH assumption. Also, there are tight secure three-round multi-signature schemes [BN05, FH21] based on this scheme. HBMS achieves the two-round signing protocol by using the special commitment scheme. Thus, in particular, we attempt to apply the technique of HBMS to the DDH-based three-round scheme to reduce the number of rounds of the signing protocol while maintaining the small reduction loss.

At first glance, although this attempt seems to work naively, it is non-trivial. This is because the special commitment used in HBMS is tailored to the multi-signature scheme based on the Schnorr signature scheme. In other words, even if we apply the special commitment scheme to the DDH-based three-round scheme, we cannot prove the security of such a scheme. Thus we need other approaches or new techniques.

To overcome this obstacle, we construct a new special commitment scheme that perfectly suits our needs. Specifically, we construct a HBMS-like special commitment scheme tailored to the DDH-based multi-signature scheme. Due to this new commitment scheme, we can construct the two-round scheme with a small reduction loss and show its security under the DDH assumption. Moreover, our commitment scheme provides a shorter signature size and smaller communication complexity rather than using the more general commitment scheme used in [PW23].

Road Maps. We first provide an overview of our techniques in Section 4.1. In Section 4.2, we show the construction of our scheme HBMSDDH-1. In Section 4.3, we show the correctness of HBMSDDH-1. After that, we explain the intuition of how we prove the security of HBMSDDH-1 under the DDH assumption in Section 4.4. In Section 4.5, we show that HBMSDDH-1 is 2-MS-UF-2 . Finally, we provide a modified scheme HBMSDDH-2 and show that HBMSDDH-2 is 2-MS-UF-1 in Section 4.6.

4.1 Technical Overview

Our goal is to construct a two-round multi-signature scheme with a small reduction loss. Our technique is based on a tightly secure DDH-based variant of the Schnorr signature scheme (i.e., a DDH-based lossy identification employed in the Katz-Wang signature scheme). Before describing our techniques in detail, we explain the difficulty of constructing a two-round multi-signature scheme from the basic Schnorr signature scheme in Section 4.1.1. Next, in order to explain the idea of our technique to construct a two-round multi-signature scheme from a tightly secure signature scheme, i.e., the Katz-Wang signature scheme, we first review the DDH-based lossy identification in Section 4.1.2. Then, we explain in detail the difficulties we face if we only naively combine already existing techniques in Section 4.1.3. Finally, we explain our solutions to overcome those difficulties in Section 4.1.4.

4.1.1 Difficulty to Construct Two-Round Schemes and Existing Techniques

Here, we explain the difficulty to construct a two-round multi-signature scheme from the Schnorr signature scheme. Schnorr signatures seem possible to be aggregated by using linearity. However, we cannot do that easily because a hash function used to sign does not have linearity. The well-known approach for this obstacle is to generate a multi-signature interactively as follows. Firstly each signer broadcasts a commitment $R_i \in \mathbb{G}$ of the

Schnorr protocol and computes $\widetilde{R} \leftarrow \sum_i R_i$. Each signer computes a challenge $c_i \in \mathbb{Z}_p$ of the Schnorr protocol by the random oracle $\mathsf{H_c}(\widetilde{R}, \mathsf{pk}_i, \mathsf{M})$, generates a response $s_i \in \mathbb{Z}_p \mod p$ of the Schnorr protocol, and sends it to all the cosigners where pk_i is a public key corresponding to the signer i and M is a message to be signed. Finally, each signer computes $\tilde{s} \leftarrow \sum_i s_i$ and outputs $(\widetilde{R}, \tilde{s})$ as a multi-signature on M. The verification equation is $\widetilde{R} = \tilde{s}G - \sum_i c_i \cdot \mathsf{pk}_i$. Unfortunately, this two-round multi-signature scheme is insecure. In this case, honest verifier zero-knowledge does not work because the reduction needs to return R to a forger before deciding c in the random oracle. There are attacks [BLL+21, DEF+19] against this multi-signature scheme.

As a solution to the above problem, Drijvers et al. proposed the secure two-round multi-signature scheme mBCJ by combining the Schnorr protocol and a homomorphic (special) equivocal commitment scheme. In a nutshell, all signers broadcast their homomorphic commitment T to R in the first round, and they broadcast their decommitment d and response s in the second round. The commitment key is generated *by the random oracle that takes a message as input*, e.g., $\mathsf{H_{ck}}(\mathsf{M})$. Thus, in the security proof, the reduction can embed either a binding commitment key or an equivocal commitment key into the random oracle $\mathsf{H_{ck}}(\mathsf{M})$. The reduction can simulate the honest signer without the secret key by exploiting (special) equivocability if the commitment keys corresponding to queried messages are equivocal keys. It can also extract the secret key of the honest signer due to the binding property and the special soundness of the Schnorr protocol if the commitment key corresponding to the forgery is a binding key.

Bellare and Dai proposed a more efficient DL-based two-round multi-signature scheme HBMS than mBCJ by using a tool like the Pedersen commitment [Ped92] instead of the homomorphic equivocal commitment scheme.

4.1.2 DDH-Based Lossy Identification

As in a well-known approach to achieve tight security of (standard) signature schemes, we attempt to construct a two-round multi-signature scheme from the Katz-Wang (standard) signature scheme [GJKW07] which employs a DDH-based lossy identification.

Here, we review the DDH-based lossy identification. The secret key is $x \in \mathbb{Z}_p$ and the public key is $(X, Y) = x(G, H)^\top$, which is a Diffie-Hellman (DH) tuple. The identification protocol is as follows. At first, the prover generates and sends $(R_1, R_2)^\top = r(G, H)^\top \in \mathbb{G}^2$ to the verifier where r is uniformly chosen from $\mathbb{Z}_p$. Next, the verifier uniformly chooses $c \xleftarrow{\$} \mathbb{Z}_p$ and sends it to the prover. After that, the prover computes $s \leftarrow r+cx \mod p$ and sends it to the verifier. Finally, the verifier checks $R = s(G,H)^\top - c(X,Y)^\top$.

The soundness is proven under the DDH assumption as follows: First, we prove that impersonation is *statistically* hard under the *lossy key* which is a non-DH tuple, i.e., there exists no $x \in \mathbb{Z}_p$ s.t. $(X,Y) = x(G,H)^\top$. Namely, under the lossy key, even for a computationally unbounded adversary, the success probability of an adversary is negligible.[1] Next, assume that there is an adversary that can perform impersonation with a non-negligible probability under a real public key, i.e., a DH tuple. Notice that this assumption induces a non-negligible gap between the adversary's success probability under a lossy key and that under a real public key. Based on this, we can construct an algorithm solving the DDH problem by internally running (without rewinding) an adversary given an instance of the DDH problem as a public key.

4.1.3 Naive Approach and Difficulty

To construct a two-round scheme with a small reduction loss, we attempt to combine the technique of HBMS and the DDH-based lossy identification. In the signing protocol of HBMS, for a commitment key $\mathsf{ck} \in \mathbb{G}$, each signer generates a commitment T by $T \leftarrow d \cdot \mathsf{ck} + R$, where d is a randomness for the commitment. Then, the verification equation is $T = d \cdot \mathsf{ck} + s \cdot G - c \cdot \mathsf{pk}$.[2] Note that one having the discrete logarithm of ck can extract the secret key from two forgeries $(T, c, (d, s))$ and $(T', c', (d', s'))$ s.t. $T = T'$ and $c \neq c'$ like the special soundness. Then, the binding property is no longer needed. Our observation means that we can replace the commitment scheme in mBCJ with a simpler and more efficient tool that has equivocability[3] and the above property like the special soundness.

We require a tool that has similar properties to HBMS to achieve our goal. More concretely, in our case, a tool needs to have the following two types of commitment keys. The first type **Type-1** ensures that forgery is statistically hard under this commitment key and a lossy key like the lossy identification. The second type **Type-2** has (special) equivocability.

We need to newly construct such a tool tailored to the DDH-based lossy

[1] Indeed, the probability of an adversary outputting s s.t. $R = s(G,H)^\top - c(X,Y)^\top$ after the verifier uniformly chooses c is at most $1/p$ independently of the behavior of the adversary because (s,c) is uniquely determined according to R when (G,H) and (X,Y) are linearly independent.

[2] For simplicity, we consider the case where there is only one signer.

[3] The equivocal key is generated by embedding the public key, and one having a trapdoor can simulate the honest signer without using the secret key.

identification because we cannot reuse the tool used in HBMS. In contrast to the Schnorr protocol, R of the DDH-based lossy identification consists of two group elements. Thus, we cannot just apply the tool of HBMS to the DDH-based lossy identification.

One may think that the following naive way is sufficient, but it is not true. Below, we explain why the naive way fails. Each signer generates two keys ck_1 and ck_2 of the tool used in HBMS by hashing the message and generates a commitment T_1 of R_1 and a commitment T_2 of R_2 by using ck_1 and ck_2, respectively. Then, the verification equation is $(T_1, T_2)^\top = d_1(\mathsf{ck}_1, O)^\top + d_2(O, \mathsf{ck}_2)^\top + s(G, H)^\top - c(X, Y)^\top$ where $c = H_c(T_1, T_2, \mathsf{M}, \mathsf{pk})$, and $d_1, d_2 \in \mathbb{Z}_p$. The important observation is that as long as the verification equation includes the term $d_1(\mathsf{ck}_1, O)^\top + d_2(O, \mathsf{ck}_2)^\top$, we cannot prove that forgery under the lossy key is statistically hard. Note that a forger can maliciously choose d_1 and d_2 so that $d_1(\mathsf{ck}_1, O)^\top + d_2(O, \mathsf{ck}_2)^\top$ is a non-DH tuple. This means that a computationally unbounded forger can forge by generating d_1, d_2, and s so that they cancel out the lossy key even if (X, Y) is a non-DH tuple. Therefore, we must modify the term to ensure the property of **Type-1**.

4.1.4 Our Solutions

Our solution is aggregating two terms $d_1(\mathsf{ck}_1, O)^\top$ and $d_2(O, \mathsf{ck}_2)^\top$ into $d(\mathsf{ck}_1, \mathsf{ck}_2)^\top$ and programming the random oracle to let $(\mathsf{ck}_1, \mathsf{ck}_2)$ be a random DH tuple. This programming is validated by the DDH assumption. When $(\mathsf{ck}_1, \mathsf{ck}_2)$ is a DH tuple, we can rewrite $(\mathsf{ck}_1, \mathsf{ck}_2)^\top = a(G, H)^\top$ where $a \in \mathbb{Z}_p$, and we obtain $(T_1, T_2)^\top = (ad + s)(G, H)^\top - c(X, Y)^\top$ as the verification equation. Notice that because (G, H) and (X, Y) are linearly independent, c satisfying the equation is determined uniquely at the point when (T_1, T_2) is determined. Since c is uniformly chosen from $\mathbb{Z}_p$ by the random oracle on input (T_1, T_2), we can prove that forgery is statistically hard. Remind that $(\mathsf{ck}_1, \mathsf{ck}_2)$ is generated by the random oracle, and thus $(\mathsf{ck}_1, \mathsf{ck}_2)$ uniformly distributes over $\mathbb{G}$ in the real environment. Then, due to the DDH assumption, we can program the random oracle to output a random DH tuple in $\mathbb{G}^2$ instead of a random tuple in $\mathbb{G}^2$.

We can construct keys of **Type-2** by embedding the public key into a DH tuple. Specifically, we program the random oracle to output $(\mathsf{ck}_1, \mathsf{ck}_2)^\top \leftarrow \rho(G, H)^\top + (X, Y)^\top$ where ρ is a trapdoor uniformly chosen from $\mathbb{Z}_p$. Due to the above approach, all terms in the verification equation are DH tuples. Then, the simulator can generate (T_1, T_2) with embedded (X, Y) in the first round and can generate (d, s) by exploiting ρ and the state information in the first round, after c is determined.

We need to guarantee that the forgery is valid under the commitment key **Type-1** and embedd those two types of commitment keys into the same random oracle to make our solution work well. Then, we use the technique of the security proof of the RSA-FDH signature scheme by Coron [Cor00]. Consequently, our scheme has a reduction loss $O(Q_S)$.

4.2 Proposed Scheme $\mathsf{HBMSDDH}$-1

Below, we show the construction of our two-round multi-signature scheme $\mathsf{HBMSDDH}$-1. For the readability, our scheme is also shown in Fig. 4.1.

$\mathsf{Setup}(1^\lambda) \to \mathsf{par}$**.** The public parameter generation algorithm takes as input the security parameter 1^λ. It sets up $(\mathbb{G}, p, G)$ by GrGen. It chooses a random element $H \in \mathbb{G}$, hash functions $\mathsf{H_c} : \{0,1\}^* \to \mathbb{Z}_p$, $\mathsf{H_{ck}} : \{0,1\}^* \to \mathbb{G}^2$, and $\mathsf{H_{agg}} : \{0,1\}^* \to \mathbb{Z}_p$, and then it outputs $\mathsf{par} = (\mathbb{G}, p, G, H, \mathsf{H_c}, \mathsf{H_{ck}}, \mathsf{H_{agg}})$.

$\mathsf{KeyGen}(\mathsf{par}) \to (\mathsf{pk}, \mathsf{sk})$**.** The key generation algorithm takes as input par, chooses $x \xleftarrow{\$} \mathbb{Z}_p$, computes $(X, Y)^\top \leftarrow x(G, H)^\top$ and outputs a public key $\mathsf{pk} = (X, Y)$ and a secret key $\mathsf{sk} = x$.

$\mathsf{Sign}^{(2)} = (\mathsf{Sign}_1^{(2)}, \mathsf{Sign}_2^{(2)})$**.** The signing protocol of our scheme consists of the following two algorithms. Let N be the number of signers, namely, $N = |\mathsf{vkList}|$.

$\mathsf{Sign}_1^{(2)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i) \to (\mathsf{pm}_i, \mathsf{st}_i)$**.** The signing algorithm for the first round takes as inputs a security parameter par, a public key list $\mathsf{vkList} = (\mathsf{pk}_j)_{j \in [N]}$, a message M to be signed, an index i of the signer, and a secret key sk_i of signer i. It parses $(X_j, Y_j)_{j \in [N]}$ from vkList, computes $t_j \leftarrow \mathsf{H_{agg}}((X_j, Y_j), \mathsf{vkList})$ for all $j \in [N]$ and $\widetilde{\mathsf{pk}} \leftarrow \sum_{j=1}^{N} t_j (X_j, Y_j)^\top$. It computes $(U_1, U_2) \leftarrow \mathsf{H_{ck}}(\mathsf{M})$, chooses $r_i, d_i \xleftarrow{\$} \mathbb{Z}_p$ and computes $T_i \leftarrow d_i (U_1, U_2)^\top + r_i (G, H)^\top$. It outputs a protocol message $\mathsf{pm}_i = T_i$ and a state $\mathsf{st}_i = (r_i, d_i, t_i, T_i, \widetilde{\mathsf{pk}})$.

$\mathsf{Sign}_2^{(2)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_j)_{j \in [N] \setminus \{i\}}) \to \mathsf{psig}_i$**.** The signing algorithm for the second round takes as input a public parameter par, a public key list vkList, a message M to be signed, an index i of the signer, a secret key sk_i and a state st_i of signer i, and a tuple of protocol messages $(\mathsf{pm}_j)_{j \in [N] \setminus \{i\}}$. It parses $(T_j)_{j \in [N] \setminus \{i\}}$ from $(\mathsf{pm}_j)_{j \in [N] \setminus \{i\}}$ and $(r_i, d_i, t_i, T_i, \widetilde{\mathsf{pk}})$ from st_i. It computes

$\widetilde{T} \leftarrow \sum_{j=1}^{N} T_j$, $c \leftarrow \mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})$, and $s_i \leftarrow x_i t_i c + r_i \mod p$. It outputs $\mathsf{psig}_i = (d_i, s_i)$.

$\mathsf{Agg}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, (\mathsf{pm}_i, \mathsf{psig}_i)_{i\in[N]}) \rightarrow \widetilde{\mathsf{sig}}$. The aggregation algorithm takes as input a public parameter par, a public key list vkList, a message M to be signed, and all signers' protocol messages and partial signatures $(\mathsf{pm}_i, \mathsf{psig}_i)_{i\in[N]}$. It parses $(X_i, Y_i)_{i\in[N]}$ from vkList and $(T_i, d_i, s_i)_{i\in[N]}$ from $(\mathsf{pm}_i, \mathsf{psig}_i)_{i\in[N]}$. It computes $t_i \leftarrow \mathsf{H_{agg}}((X_i, Y_i), \mathsf{vkList})$ for all $i \in [N]$, $\widetilde{\mathsf{pk}} \leftarrow \sum_{i=1}^{N} t_i (X_i, Y_i)^\top$, $\widetilde{T} \leftarrow \sum_{i=1}^{N} T_i$, $c \leftarrow \mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})$, $\tilde{d} \leftarrow \sum_{i=1}^{N} d_i \mod p$, and $\tilde{s} \leftarrow \sum_{i=1}^{N} s_i \mod p$ and outputs $\widetilde{\mathsf{sig}} = (c, \tilde{d}, \tilde{s})$.

$\mathsf{Verify}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, \widetilde{\mathsf{sig}}) \rightarrow \{0, 1\}$. The verification algorithm takes as inputs a public parameter par, a public key list vkList, a message M to be signed, and a multi-signature $\widetilde{\mathsf{sig}}$. It parses $(X_i, Y_i)_{i\in[N]}$ from vkList and $(c, \tilde{d}, \tilde{s})$ from $\widetilde{\mathsf{sig}}$. It computes $t_i \leftarrow \mathsf{H_{agg}}((X_i, Y_i), \mathsf{vkList})$ for all $i \in [N]$, $\widetilde{\mathsf{pk}} \leftarrow \sum_{i=1}^{N} t_i (X_i, Y_i)^\top$, $(U_1, U_2) \leftarrow \mathsf{H_{ck}}(\mathsf{M})$, $\widetilde{T} \leftarrow \tilde{d}(U_1, U_2)^\top + \tilde{s}(G, H)^\top - c \cdot \widetilde{\mathsf{pk}}$. It outputs 1 if $c = \mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})$ holds. Otherwise, it outputs 0.

Remark 3. *The aggregation algorithm can be executed by anyone who knows the public information required to run this algorithm, even if they did not participate in the signing protocol. To capture this situation, our aggregation algorithm computes the aggregated key, the aggregated commitment $\widetilde{T}$, and the challenge c. However, if the aggregation algorithm is executed by one who participated in the signing protocol, it no longer computes them since it has already computed them in the signing protocol. This means that it can efficiently aggregate the partial signatures to produce the multi-signature. We implement our scheme for this situation in Section 5.2.*

$\mathsf{Setup}(1^\lambda)$:

$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$

$H \xleftarrow{\$} \mathbb{G}$

Select $\mathsf{H_c}$ // $\mathsf{H_c} : \{0,1\}^* \to \mathbb{Z}_p$

Select $\mathsf{H_{ck}}$ // $\mathsf{H_{ck}} : \{0,1\}^* \to \mathbb{G}^2$

Select $\mathsf{H_{agg}}$ // $\mathsf{H_{agg}} : \{0,1\}^* \to \mathbb{Z}_p$

$\mathsf{par} \leftarrow (\mathbb{G}, p, G, H, \mathsf{H_c}, \mathsf{H_{ck}}, \mathsf{H_{agg}})$

return par

$\mathsf{KeyGen}(\mathsf{par}) \to (\mathsf{pk}, \mathsf{sk})$:

$x \xleftarrow{\$} \mathbb{Z}_p$

$(X, Y)^\top \leftarrow x(G, H)^\top$

$\mathsf{pk} \leftarrow (X, Y)$

$\mathsf{sk} \leftarrow x$

return $(\mathsf{pk}, \mathsf{sk})$

$\mathsf{Agg}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, (\mathsf{pm}_i, \mathsf{psig}_i)_{i\in[N]})$:

parse $(X_i, Y_i)_{i\in[N]} \leftarrow \mathsf{vkList}$

parse $(T_i, d_i, s_i)_{i\in[N]}$
$\qquad \leftarrow (\mathsf{pm}_i, \mathsf{psig}_i)_{i\in[N]}$

for $i \in [N]$ **do**

$\quad t_i \leftarrow \mathsf{H_{agg}}((X_i, Y_i), \mathsf{vkList})$

$\widetilde{\mathsf{pk}} \leftarrow \sum_{i=1}^{N} t_i (X_i, Y_i)^\top$

$\widetilde{T} \leftarrow \sum_{i=1}^{N} T_i$

$c \leftarrow \mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})$

$\tilde{d} \leftarrow \sum_{i=1}^{N} d_i \mod p$

$\tilde{s} \leftarrow \sum_{i=1}^{N} s_i \mod p$

$\widetilde{\mathsf{sig}} \leftarrow (c, \tilde{d}, \tilde{s})$

return $\widetilde{\mathsf{sig}}$

$\mathsf{Sign}_1^{(2)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i)$:

parse $(X_j, Y_j)_{j\in[N]} \leftarrow \mathsf{vkList}$

for $j \in [N]$ **do** $t_j \leftarrow \mathsf{H_{agg}}((X_j, Y_j), \mathsf{vkList})$

$\widetilde{\mathsf{pk}} \leftarrow \sum_{j=1}^{N} t_j (X_j, Y_j)^\top$

$(U_1, U_2) \leftarrow \mathsf{H_{ck}}(\mathsf{M})$

$r_i, d_i \xleftarrow{\$} \mathbb{Z}_p$

$T_i \leftarrow d_i (U_1, U_2)^\top + r_i (G, H)^\top$

$\mathsf{pm}_i \leftarrow T_i$

$\mathsf{st}_i \leftarrow (r_i, d_i, t_i, T_i, \widetilde{\mathsf{pk}})$

return $(\mathsf{pm}_i, \mathsf{st}_i)$

$\mathsf{Sign}_2^{(2)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_j)_{j\in[N]\setminus\{i\}})$:

parse $(T_j)_{j\in[N]\setminus\{i\}} \leftarrow (\mathsf{pm}_j)_{j\in[N]\setminus\{i\}}$

parse $(r_i, d_i, t_i, T_i, \widetilde{\mathsf{pk}}) \leftarrow \mathsf{st}_i$

$\widetilde{T} \leftarrow \sum_{j=1}^{N} T_j$

$c \leftarrow \mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})$

$s_i \leftarrow x_i t_i c + r_i \mod p$

$\mathsf{psig}_i \leftarrow (d_i, s_i)$

return psig_i

$\mathsf{Verify}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, \widetilde{\mathsf{sig}})$:

parse $(X_i, Y_i)_{i\in[N]} \leftarrow \mathsf{vkList}$

parse $(c, \tilde{d}, \tilde{s}) \leftarrow \widetilde{\mathsf{sig}}$

for $i \in [N]$ **do** $t_i \leftarrow \mathsf{H_{agg}}((X_i, Y_i), \mathsf{vkList})$

$\widetilde{\mathsf{pk}} \leftarrow \sum_{i=1}^{N} t_i (X_i, Y_i)^\top$

$(U_1, U_2) \leftarrow \mathsf{H_{ck}}(\mathsf{M})$

$\widetilde{T} \leftarrow \tilde{d}(U_1, U_2)^\top + \tilde{s}(G, H)^\top - c \cdot \widetilde{\mathsf{pk}}$

if $[\![c = \mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})]\!]$ **then**

$\quad$ **return** 1

return 0

Figure 4.1: The construction of HBMSDDH-1. $\mathsf{H_c}, \mathsf{H_{ck}}$, and $\mathsf{H_{agg}}$ are modeled as random oracles. N is the number of the signers in vkList.

4.3 Correctness of HBMSDDH-1

Here, we show that our scheme is correct.

Theorem 12. HBMSDDH-1 *in Section 4.2 (and in Fig. 4.1) satisfies correctness.*

Proof. If all signers participated in the signing protocol honestly executes the signing protocol on M and vkList, then we have

$$\begin{aligned}
&\tilde{d}(U_1, U_2)^\top + \tilde{s}(G, H)^\top - c \cdot \widetilde{\mathsf{pk}} \\
&= \sum_{i=1}^{N} d_i(U_1, U_2)^\top + \sum_{i=1}^{N} s_i(G, H)^\top - c \cdot \sum_{i=1}^{N} t_i x_i (G, H)^\top \\
&= \sum_{i=1}^{N} d_i(U_1, U_2)^\top + \sum_{i=1}^{N} (x_i t_i c + r_i)(G, H)^\top - c \cdot \sum_{i=1}^{N} t_i x_i (G, H)^\top \\
&= \sum_{i=1}^{N} d_i(U_1, U_2)^\top + r_i(G, H)^\top = \sum_{i=1}^{N} T_i = \widetilde{T}.
\end{aligned}$$

Therefore, our scheme satisfies correctness. □

4.4 Intuition of Security Proof

Before showing the full security proof, we show a proof sketch.

As in the Katz-Wang signature scheme, we prove the unforgeability of HBMSDDH-1 by replacing the public key with a non-DH tuple due to the DDH assumption and proving that forgery is statistically hard under such a public key. The main strategy is that we ensure a situation where we can statistically evaluate the adversary's success probability $\epsilon_{\mathcal{A}}$, namely, we can show that $\epsilon_{\mathcal{A}}$ is negligible even if the adversary 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 $\mathsf{H}_{\mathsf{ck}}(\mathsf{M})$ 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 [Cor00], which is to prove the security of the RSA Full Domain Hash (RSA-FDH) signature scheme [BR93], 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 **(Game_1-Game_3):** 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 $\mathsf{H}_{\mathsf{ck}}(\mathsf{M})$ according to a biased coin which comes out heads with a certain probability, like the technique of Coron [Cor00]. The first type (**Type-1**) is to statistically evaluate the success probability of an adversary in the final game. The other type (**Type-2**) is to simulate the honest signer without sk in the signing oracle.

Game G_2 **(Game_4):** 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 **Type-2**.

Game G_3 **(Game_5-Game_6) :** 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, first, we show our procedure to prove that G_1 and G_3 are computationally indistinguishable under the DDH assumption. First, we prove that G_1 and 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 **Type-2** are perfectly indistinguishable. Next, we prove that G_2 and G_3 are computationally indistinguishable by proving that pk generated by KeyGen in G_2 and G_3 which are a DH tuple and a non-DH tuple are indistinguishable under the DDH assumption.

Using the property of (U_1, U_2) of **Type-1**, in Game G_3, we can statistically evaluate $\epsilon_{\mathcal{A}}$ 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 **Type-1**. The challenger generates it satisfying that it is uniformly distributed in the span of (G, H). Specifically, the challenger chooses $\rho \xleftarrow{\$} \mathbb{Z}_p$ and computes $(U_1, U_2)^\top \leftarrow \rho(G, H)^\top$. 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 $T_1 = d_1(U_1, U_2)^\top + s_1(G, H)^\top - c(X_1, Y_1)^\top$ where $c = \mathsf{H}_{\mathsf{c}}(T_1, (X_1, Y_1), \mathsf{M})$. Notice that, when (G, H, X_1, Y_1) is a non-DH tuple and (U_1, U_2) is in the span of (G, H), c satisfying the above equation is determined uniquely at the point when T_1 is determined. Because c is uniformly chosen from $\mathbb{Z}_p$ by the random oracle, the probability that c satisfies the above equation is at most $1/p$.[4]

Next, we explain the way to generate (U_1, U_2) of **Type-2**. The challenge key (X_1, Y_1) is embedded in this type of (U_1, U_2). More concretely, the challenger chooses $\rho \xleftarrow{\$} \mathbb{Z}_p$ and computes $(U_1, U_2)^\top \leftarrow \rho(G, H)^\top + (X_1, Y_1)^\top$. Then, the challenger has ρ as the trapdoor. The equivocal commitment T_1 is generated by $\alpha(G, H)^\top + \beta(X_1, Y_1)^\top$ where α and β are uniformly chosen from $\mathbb{Z}_p$. We need to produce d_1 and s_1 satisfying $T_1 = d_1(U_1, U_2)^\top + s_1(G, H)^\top - c(X_1, Y_1)^\top$ given c. Notice that sk is no longer required since T_1 and $(U_1, U_2)^\top$ is expressed by linear combinations of $(G, H)^\top$ and $(X_1, Y_1)^\top$. Specifically, by using ρ, α, and β, we can produce d_1 and s_1 satisfying $T_1 = \alpha(G, H)^\top + \beta(X_1, Y_1)^\top = d_1(\rho(G, H)^\top + (X_1, Y_1)^\top) + s_1(G, H)^\top - c(X_1, Y_1)^\top$. For indistinguishability between the distribution of responses of the signing oracle with sk and that of responses of the signing oracle using ρ, see Lemma 6.

Finally, we explain that the game of the security definition and G_1 are computationally indistinguishable under the DDH assumption. Notice that, for both types of (U_1, U_2), the challenger generates (U_1, U_2) by producing a random DH tuple $\rho(G, H)^\top$. 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 $\mathsf{H}_{\mathsf{ck}}(\mathsf{M})$ in G_1 are computationally indistinguishable under the DDH assumption. Therefore, the game of the security definition and G_1 are computationally indistinguishable under the DDH assumption.

As a result, we can show that $\mathsf{HBMSDDH\text{-}1}$ is secure under the DDH assumption in the random oracle model.

Remark 4. *Since we need to guarantee that the adversary only produces forgery which is valid under (U_1, U_2) of **Type-1**, we need to add a condition that a forgery is valid under (U_1, U_2) of **Type-1** into the winning condition of the adversary in G_1. In the full proof, we consider intermediate games between the original game of the security definition and G_1 with the above additional winning condition. Thus, $\mathsf{HBMSDDH\text{-}1}$ 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 [Cor00].*

[4]Because our scheme supports key aggregation, we need to consider a more complex setting. For more details, see Lemma 8.

4.5 Formal Security Proof for $\mathsf{HBMSDDH\text{-}1}$

Theorem 13. *If $\mathbb{G}$ is a (t, ϵ)-DDH group, then* $\mathsf{HBMSDDH\text{-}1}$ *is $(t_{\mathcal{A}}, Q_S, Q_H, N, \epsilon_{\mathcal{A}})$-2-MS-UF-2 s.t.*

$$\begin{aligned}
&\epsilon_{\mathcal{A}} \geq e(Q_S+1)\left(2\epsilon + \frac{2Q_H + Q_S + 2}{p}\right) \text{ and} \\
&t_{\mathcal{A}} \leq \min(t_1, t_2) \text{ where} \\
&t_1 = t - (4Q_H + 6Q_SN + 4Q_S + 2N + 12)t_{\mathrm{mul}} - O(Q_H + Q_SN), \\
&t_2 = t - (2Q_H + 6Q_SN + 2Q_S + 2N + 8)t_{\mathrm{mul}} - O(Q_H + Q_SN),
\end{aligned}$$

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

Proof of Theorem 14. First, we prepare two notations. Let us denote the time for a scalar multiplication in $\mathbb{G}$ by t_{mul}. We write $\Pr[\mathsf{Game}_i = 1]$ to mean the probability that a forger wins the game Game_i.

Let $\mathcal{A}$ be an adversary that $(t_{\mathcal{A}}, Q_S, Q_H, N, \epsilon_{\mathcal{A}})$-breaks the 2-MS-UF-2 of $\mathsf{HBMSDDH\text{-}1}$. For $\mathcal{A}$, we consider a sequence of games where the first hybrid Game_0 is the slightly weak unforgeability game in Fig. 2.10 for $\mathsf{HBMSDDH\text{-}1}$. Game_0 is shown in Fig. 4.2. Because Game_0 is the unforgeability game of $\mathsf{HBMSDDH\text{-}1}$, we have $\Pr[\mathsf{Game}_0 = 1] \geq \epsilon_{\mathcal{A}}$.

Now we change Game_0 as follows.

$\underline{\mathsf{Game}_1}$**:** In this game, the challenger makes a query $\mathsf{H}_{\mathsf{ck}}(\mathsf{M})$ at the beginning of both the random oracle H_{c} and the signing oracle $\mathcal{O}_{\mathsf{Sign}_1^{(2)}}$. The changed game is depicted in Fig. 4.3. These newly added steps do not affect the probability of $\mathcal{A}$ winning the game. Therefore, we have

$$\Pr[\mathsf{Game}_0 = 1] = \Pr[\mathsf{Game}_1 = 1].$$

$\underline{\mathsf{Game}_2}$**:** In this game, the challenger partition the outputs (U_1, U_2) of $\mathsf{H}_{\mathsf{ck}}(\mathsf{M})$ into two groups following a biased coin $b_K \in \{0, 1\}$ and aborts the game if a bit b_K corresponding to (U_1, U_2) used in a signing query is 0 at the beginning of the second round. Moreover, it additionally checks the condition $\mathsf{T}_{\mathsf{b}}[\mathsf{M}^*] = 0$. The changed game is depicted in Fig. 4.4. Specifically, it additionally initializes a table $\mathsf{T}_{\mathsf{b}}[\cdot] \leftarrow \perp$ at the beginning of the game. It firstly chooses a bit $b_K \in \{0, 1\}$ which becomes 1 with probability $\delta = Q_S/(Q_S + 1)$ and assigns $\mathsf{T}_{\mathsf{b}}[\mathsf{M}] \leftarrow b_K$. Note that the way to generate (U_1, U_2) is unchanged. In the signing oracle $\mathcal{O}_{\mathsf{Sign}_2^{(2)}}$, it aborts the game if $\mathsf{T}_{\mathsf{b}}[\mathsf{M}] = 0$ holds. Otherwise, it continues the game.

$\mathsf{Game}_0 = \mathsf{Game}^{\mathsf{ms^2\text{-}uf2}}_{\mathsf{HBMSDDH\text{-}1},\mathcal{A}}(1^\lambda, N)$

$\mathsf{Q_M} \leftarrow \emptyset, \mathsf{Q_{st}}[\cdot] \leftarrow \bot$
$\mathsf{T_{H_c}}[\cdot] \leftarrow \bot, \mathsf{T_{H_{ck}}}[\cdot] \leftarrow \bot, \mathsf{T_{H_{agg}}}[\cdot] \leftarrow \bot$
$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$
$H \xleftarrow{\$} \mathbb{G}$
$x \xleftarrow{\$} \mathbb{Z}_p$
$(X, Y)^\top \leftarrow x(G, H)^\top$
$\mathsf{pk} \leftarrow (X, Y)$
$\mathsf{sk} \leftarrow x$
$(\mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}_1^{(2)}}, \mathcal{O}_{\mathsf{Sign}_2^{(2)}}, \mathsf{H_c}, \mathsf{H_{ck}}, \mathsf{H_{agg}}}(\mathsf{par}, \mathsf{pk})$
req $[\![\mathsf{pk} \in \mathsf{vkList}^*]\!] \wedge [\![|\mathsf{vkList}^*| \leq N]\!] \wedge [\![(\mathsf{M}^*) \notin \mathsf{Q_M}]\!]$
return $\mathsf{Verify}(\mathsf{par}, \mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*)$

$\mathcal{O}_{\mathsf{Sign}_1^{(2)}}(\mathsf{sid}, \mathsf{vkList}, \mathsf{M})$

req $[\![\mathsf{pk} \in \mathsf{vkList}]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 1] = \bot]\!] \wedge [\![|\mathsf{vkList}| \leq N]\!]$
$\mathsf{HS_{sid}} \leftarrow \emptyset$
$N \leftarrow |\mathsf{vkList}|$
parse $(\mathsf{pk}_i)_{i \in [N]} \leftarrow \mathsf{vkList}$
for $i \in [N]$ **do**
  if $\mathsf{pk}_i = \mathsf{pk}$ **then**
    $\mathsf{HS_{sid}} \leftarrow \mathsf{HS_{sid}} \cup \{i\}$
parse $(X_j, Y_j)_{j \in [N]} \leftarrow \mathsf{vkList}$
for $j \in [N]$ **do**
  $t_j \leftarrow \mathsf{H_{agg}}((X_j, Y_j), \mathsf{vkList})$
$\widetilde{\mathsf{pk}} \leftarrow \sum_{j=1}^{N} t_j (X_j, Y_j)^\top$
$(U_1, U_2) \leftarrow \mathsf{H_{ck}}(\mathsf{M})$
for $i \in \mathsf{HS_{sid}}$ **do**
  $r_i, d_i \xleftarrow{\$} \mathbb{Z}_p$
  $T_i \leftarrow d_i (U_1, U_2)^\top + r_i (G, H)^\top$
  $\mathsf{pm}_i \leftarrow T_i$
  $\mathsf{st}_i \leftarrow (r_i, d_i, t_i, T_i, \widetilde{\mathsf{pk}})$
$\mathsf{Q_{st}}[\mathsf{sid}, 1] \xleftarrow{\$} (\mathsf{vkList}, \mathsf{M}, \mathsf{HS_{sid}}, (\mathsf{st}_i)_{i \in \mathsf{HS_{sid}}})$
return $(\mathsf{pm}_i)_{i \in \mathsf{HS_{sid}}}$

$\mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})$:

if $[\![\mathsf{T_{H_c}}[\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M}] = \bot]\!]$
  $c \xleftarrow{\$} \mathbb{Z}_p$
  $\mathsf{T_{H_c}}[\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M}] \leftarrow c$
return $\mathsf{T_{H_c}}[\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M}]$

$\mathsf{H_{ck}}(\mathsf{M})$:

if $[\![\mathsf{T_{H_{ck}}}[\mathsf{M}] = \bot]\!]$
  $(U_1, U_2) \xleftarrow{\$} \mathbb{G}^2$
  $\mathsf{T_{H_{ck}}}[\mathsf{M}] \leftarrow (U_1, U_2)$
return $\mathsf{T_{H_{ck}}}[\mathsf{M}]$

$\mathsf{H_{agg}}((X, Y), \mathsf{vkList})$:

if $[\![\mathsf{T_{H_{agg}}}[(X, Y), \mathsf{vkList}] = \bot]\!]$
  $t \xleftarrow{\$} \mathbb{Z}_p$
  $\mathsf{T_{H_{agg}}}[(X, Y), \mathsf{vkList}] \leftarrow t$
return $\mathsf{T_{H_{agg}}}[(X, Y), \mathsf{vkList}]$

$\mathcal{O}_{\mathsf{Sign}_2^{(2)}}(\mathsf{sid}, (\mathsf{pm}_j)_{j \in [|\mathsf{vkList}|] \setminus \mathsf{HS}})$

req $[\![\mathsf{Q_{st}}[\mathsf{sid}, 1] \neq \bot]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 2] = \bot]\!]$
$(\mathsf{vkList}, \mathsf{M}, \mathsf{HS_{sid}}, (\mathsf{st}_i)_{i \in \mathsf{HS_{sid}}}) \leftarrow \mathsf{Q_{st}}[\mathsf{sid}, 1]$
$N \leftarrow |\mathsf{vkList}|$
parse $(T_j)_{j \in [N] \setminus \mathsf{HS}} \leftarrow (\mathsf{pm}_j)_{j \in [N] \setminus \mathsf{HS}}$
for $i \in \mathsf{HS_{sid}}$ **do**
  parse $(r_i, d_i, t_i, T_i, \widetilde{\mathsf{pk}}) \leftarrow \mathsf{st}_i$
$\widetilde{T} \leftarrow \sum_{j=1}^{N} T_j$
$c \leftarrow \mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})$
parse $x \leftarrow \mathsf{sk}$
for $i \in \mathsf{HS_{sid}}$ **do**
  $s_i \leftarrow x t_i c + r_i \mod p$
  $\mathsf{psig}_i \leftarrow (d_i, s_i)$
$\mathsf{Q_{st}}[\mathsf{sid}, 2] \leftarrow (\mathsf{psig}_i)_{i \in \mathsf{HS_{sid}}}$
$\mathsf{Q_M} \leftarrow \mathsf{Q_M} \cup \{\mathsf{M}\}$
return $(\mathsf{psig}_i)_{i \in \mathsf{HS_{sid}}}$

Figure 4.2: The initial game Game_0, that identical to the slightly weak unforgeability game in Fig. 2.10 for HBMSDDH-1.

At the end of the game, it checks the condition $\mathsf{T_b}[\mathsf{M}^*] = 0$ in addition to other conditions. Due to Lemma 2, which we will prove later, we have

$$\Pr[\mathsf{Game}_1 = 1] \leq e(Q_S + 1) \Pr[\mathsf{Game}_2 = 1].$$

Game_1:

$\mathcal{O}_{\mathsf{Sign}_1^{(2)}}(\mathsf{sid}, \mathsf{vkList}, \mathsf{M})$

$\textbf{req}\ [\![\mathsf{pk} \in \mathsf{vkList}]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 1] = \bot]\!] \wedge [\![|\mathsf{vkList}| \leq N]\!]$
$(U_1, U_2) \leftarrow \mathsf{H_{ck}}(\mathsf{M})$
// Identical to Lines 2 to 18 of $\mathcal{O}_{\mathsf{Sign}_1^{(2)}}$ in Game_0.

$\mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})$:

$(U_1, U_2) \leftarrow \mathsf{H_{ck}}(\mathsf{M})$
$\textbf{if}\ \left[\!\!\left[\mathsf{T_{H_c}}[\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M}] = \bot\right]\!\!\right]$
$\quad c \xleftarrow{\$} \mathbb{Z}_p$
$\quad \mathsf{T_{H_c}}[\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M}] \leftarrow c$
$\textbf{return}\ \mathsf{T_{H_c}}[\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M}]$

Figure 4.3: The first game Game_1. The changes from Game_0 are highlighted in blue. For readability, we omit the lines of $\mathcal{O}_{\mathsf{Sign}_1^{(2)}}$ that are identical to those of $\mathcal{O}_{\mathsf{Sign}_1^{(2)}}$ in Game_0.

Game_2, $\boxed{\mathsf{Game}_3}$

$\mathsf{Q_M} \leftarrow \emptyset, \mathsf{Q_{st}}[\cdot] \leftarrow \bot$
$\mathsf{T_{H_c}}[\cdot] \leftarrow \bot, \mathsf{T_{H_{ck}}}[\cdot] \leftarrow \bot, \mathsf{T_{H_{agg}}}[\cdot] \leftarrow \bot$
$\mathsf{T_b}[\cdot] \leftarrow \bot$, $\boxed{\mathsf{T_{td}}[\cdot] \leftarrow \bot}$
$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$
$H \xleftarrow{\$} \mathbb{G}$
$x \xleftarrow{\$} \mathbb{Z}_p$
$(X, Y)^\top \leftarrow x(G, H)^\top$
$\mathsf{pk} \leftarrow (X, Y)$
$\mathsf{sk} \leftarrow x$
$(\mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}_1^{(2)}}, \mathcal{O}_{\mathsf{Sign}_2^{(2)}}, \mathsf{H_c}, \mathsf{H_{ck}}, \mathsf{H_{agg}}}(\mathsf{par}, \mathsf{pk})$
$\textbf{req}\ [\![\mathsf{pk} \in \mathsf{vkList}^*]\!] \wedge [\![|\mathsf{vkList}^*| \leq N]\!] \wedge [\![(\mathsf{M}^*) \notin \mathsf{Q_M}]\!]$
$\textbf{req}\ [\![\mathsf{T_b}[\mathsf{M}^*] = 0]\!]$
$\textbf{return}\ \mathsf{Verify}(\mathsf{par}, \mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*)$

$\mathcal{O}_{\mathsf{Sign}_2^{(2)}}(\mathsf{sid}, (\mathsf{pm}_j)_{j \in [|\mathsf{vkList}|] \setminus \mathsf{HS}})$

$\textbf{req}\ [\![\mathsf{Q_{st}}[\mathsf{sid}, 1] \neq \bot]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 2] = \bot]\!]$
$\textbf{if}\ [\![\mathsf{T_b}[\mathsf{M}] = 0]\!]\ \textbf{then}$
$\quad \textbf{abort}$
// Identical to Lines 2 to 15 of $\mathcal{O}_{\mathsf{Sign}_2^{(2)}}$ in Game_0

$\mathsf{H_{ck}}(\mathsf{M})$:

$\textbf{if}\ [\![\mathsf{T_{H_{ck}}}[\mathsf{M}] = \bot]\!]$
$\quad b_K = 0$
$\quad b_K \leftarrow 1$ with probability $\delta = Q_S/(Q_S + 1)$
$\quad \mathsf{T_b}[\mathsf{M}] \leftarrow b_K$
$\quad (U_1, U_2) \xleftarrow{\$} \mathbb{G}^2$ // For Game_2.
$\quad \boxed{\rho \xleftarrow{\$} \mathbb{Z}_p}$
$\quad \boxed{\textbf{if}\ [\![\mathsf{T_b}[\mathsf{M}] = 0]\!]\ \textbf{then}}$
$\quad \boxed{\quad (U_1, U_2)^\top \leftarrow \rho(G, H)^\top}$
$\quad \boxed{\textbf{else}}$
$\quad \boxed{\quad (U_1, U_2)^\top \leftarrow \rho(G, H)^\top + (X, Y)^\top}$
$\quad \boxed{\mathsf{T_{td}}[\mathsf{M}] \leftarrow \rho}$
$\quad \mathsf{T_{H_{ck}}}[\mathsf{M}] \leftarrow (U_1, U_2)$
$\textbf{return}\ \mathsf{T_{H_{ck}}}[\mathsf{M}]$

Figure 4.4: The second game Game_2 and the third game Game_3. The changes from Game_1 and Game_2 are highlighted in blue and boxed, respectively. For readability, we omit the lines of $\mathcal{O}_{\mathsf{Sign}_2^{(2)}}$ that are identical to those of $\mathcal{O}_{\mathsf{Sign}_2^{(2)}}$ in Game_0.

$\underline{\mathsf{Game}_3}$**:** In this game, the challenger modifies how it generates (U_1, U_2) in $\mathsf{H}_{\mathsf{ck}}(\mathsf{M})$ corresponding to a value $\mathsf{T}_{\mathsf{b}}[\mathsf{M}]$. The changed game is depicted in Fig. 4.4. Specifically, it additionally initializes a table $\mathsf{T}_{\mathrm{td}}[\cdot] \leftarrow \bot$ at the beginning of the game. In H_{ck}, instead of $(U_1, U_2) \xleftarrow{\$} \mathbb{G}^2$, it chooses $\rho \xleftarrow{\$} \mathbb{Z}_p$, computes $(U_1, U_2)^\top \xleftarrow{\$} \rho(G, H)^\top$ when $\mathsf{T}_{\mathsf{b}}[\mathsf{M}] = 0$, and computes $(U_1, U_2)^\top \xleftarrow{\$} \rho(G, H)^\top + (X, Y)^\top$ when $\mathsf{T}_{\mathsf{b}}[\mathsf{M}] = 1$. Then, it assigns $\mathsf{T}_{\mathrm{td}}[\mathsf{M}] \leftarrow \rho$. Due to Lemma 3, which we will prove later, assuming that $\mathbb{G}$ is a (t, ϵ)-DDH group, for $\mathcal{A}$ such that $t_{\mathcal{A}} \leq t - (4Q_H + 6Q_S N + 4Q_S + 2N + 12)t_{\mathrm{mul}} - O(Q_H + Q_S N)$, we have

$$|\Pr[\mathsf{Game}_2 = 1] - \Pr[\mathsf{Game}_3 = 1]| \leq \epsilon.$$

$\underline{\mathsf{Game}_4}$**:** In this game, the challenger modifies how it generates protocol messages and partial signatures. This is depicted in Fig. 4.5. Specifically, in $\mathcal{O}_{\mathsf{Sign}_1^{(2)}}$, it chooses $(\alpha_i, \beta_i) \xleftarrow{\$} \mathbb{Z}_p^2$ and computes $T_i \leftarrow \alpha_i(G, H)^\top + \beta_i(X, Y)^\top$. In $\mathcal{O}_{\mathsf{Sign}_2^{(2)}}$, it computes $d_i \leftarrow \beta_i + c \mod p$ and $s_i \leftarrow \alpha_i - d_i\rho \mod p$ by using the trapdoor ρ corresponding to $(U_1, U_2) = \mathsf{H}_{\mathsf{ck}}(\mathsf{M})$. Due to Lemma 6, which we will prove later, we have

$$\Pr[\mathsf{Game}_3 = 1] = \Pr[\mathsf{Game}_4 = 1].$$

$\underline{\mathsf{Game}_5}$**:** In this game, the challenger changes how it generates the challenge key. This game is depicted in Fig. 4.6. Specifically, it additionally chooses $y \xleftarrow{\$} \mathbb{Z}_p \backslash \{x\}$ and computes $X \leftarrow xG$ and $Y \leftarrow yH$, instead of $(X, Y)^\top \leftarrow x(G, H)^\top$. Due to Lemma 7, which we will prove later, assuming $\mathbb{G}$ is a (t, ϵ)-DDH group, for $\mathcal{A}$ such that $t_{\mathcal{A}} \leq t - (2Q_H + 6Q_S N + 2Q_S + 2N + 8)t_{\mathrm{mul}} - O(Q_H + Q_S N)$, we have

$$|\Pr[\mathsf{Game}_4 = 1] - \Pr[\mathsf{Game}_5 = 1]| \leq \epsilon.$$

$\underline{\mathsf{Game}_6}$**:** In this game, the challenger defines $\mathsf{H}_{\mathsf{agg}}((X_i, Y_i), \mathsf{vkList})$ for all $(X_i, Y_i) \in \mathsf{vkList}$. This is depicted in Fig. 4.6. Specifically, when $((X, Y), \mathsf{vkList})$ is queried to $\mathsf{H}_{\mathsf{agg}}$, after defining $\mathsf{H}_{\mathsf{agg}}((X, Y), \mathsf{vkList})$, it chooses $t_i \xleftarrow{\$} \mathbb{Z}_p$ and assigns $\mathsf{T}_{\mathsf{H}_{\mathsf{agg}}}[(X_i, Y_i), \mathsf{vkList}] \leftarrow t_i$ for all $(X_i, Y_i) \in \mathsf{vkList}$. Since the challenger gives $\mathcal{A}$ only $\mathsf{T}_{\mathsf{H}_{\mathsf{agg}}}[(X, Y), \mathsf{vkList}]$, where $((X, Y), \mathsf{vkList})$ is queried, this change does not affect the probability of $\mathcal{A}$ winning the game. Thus, we have

$$\Pr[\mathsf{Game}_5 = 1] = \Pr[\mathsf{Game}_6 = 1].$$

Game_4:

$\mathcal{O}_{\mathsf{Sign}_1^{(2)}}(\mathsf{sid}, \mathsf{vkList}, \mathsf{M})$

$\textbf{req}\ [\![\mathsf{pk} \in \mathsf{vkList}]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 1] = \bot]\!] \wedge [\![|\mathsf{vkList}| \leq N]\!]$
$(U_1, U_2) \leftarrow \mathsf{H_{ck}}(\mathsf{M})$
$\mathsf{HS_{sid}} \leftarrow \emptyset$
$N \leftarrow |\mathsf{vkList}|$
$\textbf{parse}\ (\mathsf{pk}_i)_{i \in [N]} \leftarrow \mathsf{vkList}$
$\textbf{for}\ i \in [N]\ \textbf{do}$
$\quad \textbf{if}\ \mathsf{pk}_i = \mathsf{pk}\ \textbf{then}$
$\quad\quad \mathsf{HS_{sid}} \leftarrow \mathsf{HS_{sid}} \cup \{i\}$
$\textbf{parse}\ (X_j, Y_j)_{j \in [N]} \leftarrow \mathsf{vkList}$
$\textbf{for}\ j \in [N]\ \textbf{do}$
$\quad t_j \leftarrow \mathsf{H_{agg}}((X_j, Y_j), \mathsf{vkList})$
$\widetilde{\mathsf{pk}} \leftarrow \sum_{j=1}^{N} t_j (X_j, Y_j)^\top$
$\textbf{for}\ i \in \mathsf{HS_{sid}}\ \textbf{do}$
$\quad \alpha_i, \beta_i \xleftarrow{\$} \mathbb{Z}_p$
$\quad T_i \leftarrow \alpha_i (G, H)^\top + \beta_i (X, Y)^\top$
$\quad \mathsf{pm}_i \leftarrow T_i$
$\quad \mathsf{st}_i \leftarrow (\alpha_i, \beta_i, t_i, T_i, \widetilde{\mathsf{pk}})$
$\mathsf{Q_{st}}[\mathsf{sid}, 1] \xleftarrow{\$} (\mathsf{vkList}, \mathsf{M}, \mathsf{HS_{sid}}, (\mathsf{st}_i)_{i \in \mathsf{HS_{sid}}})$
$\textbf{return}\ (\mathsf{pm}_i)_{i \in \mathsf{HS_{sid}}}$

$\mathcal{O}_{\mathsf{Sign}_2^{(2)}}(\mathsf{sid}, (\mathsf{pm}_j)_{j \in [|\mathsf{vkList}|] \setminus \mathsf{HS}})$

$\textbf{req}\ [\![\mathsf{Q_{st}}[\mathsf{sid}, 1] \neq \bot]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 2] = \bot]\!]$
$\textbf{if}\ [\![\mathsf{T_b}[\mathsf{M}] = 0]\!]\ \textbf{then}$
$\quad \textbf{abort}$
$(\mathsf{vkList}, \mathsf{M}, \mathsf{HS_{sid}}, (\mathsf{st}_i)_{i \in \mathsf{HS_{sid}}}) \leftarrow \mathsf{Q_{st}}[\mathsf{sid}, 1]$
$N \leftarrow |\mathsf{vkList}|$
$\textbf{parse}\ (T_j)_{j \in [N] \setminus \mathsf{HS}} \leftarrow (\mathsf{pm}_j)_{j \in [N] \setminus \mathsf{HS}}$
$\textbf{for}\ i \in \mathsf{HS_{sid}}\ \textbf{do}$
$\quad \textbf{parse}\ (\alpha_i, \beta_i, t_i, T_i, \widetilde{\mathsf{pk}}) \leftarrow \mathsf{st}_i$
$\widetilde{T} \leftarrow \sum_{j=1}^{N} T_j$
$c \leftarrow \mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})$
$\rho \leftarrow \mathsf{T_{td}}[\mathsf{M}]$
$\textbf{for}\ i \in \mathsf{HS_{sid}}\ \textbf{do}$
$\quad d_i \leftarrow \beta_i + c \mod p$
$\quad s_i \leftarrow \alpha_i - d_i \rho \mod p$
$\quad \mathsf{psig}_i \leftarrow (d_i, s_i)$
$\mathsf{Q_{st}}[\mathsf{sid}, 2] \leftarrow (\mathsf{psig}_i)_{i \in \mathsf{HS_{sid}}}$
$\mathsf{Q_M} \leftarrow \mathsf{Q_M} \cup \{\mathsf{M}\}$
$\textbf{return}\ (\mathsf{psig}_i)_{i \in \mathsf{HS_{sid}}}$

Figure 4.5: The fourth game Game_4. The changes from Game_3 are highlighted in blue.

From Lemma 8, which we will prove later, the advantage of $\mathcal{A}$ against Game_6 is statistically bounded as

$$\Pr[\mathsf{Game}_6 = 1] \leq \frac{2Q_H + Q_S + 2}{p}.$$

By combining all arguments, we obtain the the following statement. If $\mathbb{G}$ is a (t, ϵ)-DDH group, for $\mathcal{A}$ such that

$$t_{\mathcal{A}} \leq t - (4Q_H + 6Q_S N + 4Q_S + 2N + 12)t_{\text{mul}} - O(Q_H + Q_S N),$$
$$\text{and } t_{\mathcal{A}} \leq t - (2Q_H + 6Q_S N + 2Q_S + 2N + 8)t_{\text{mul}} - O(Q_H + Q_S N),$$

Game_5	Game_6:

$\mathsf{Q_M} \leftarrow \emptyset, \mathsf{Q_{st}}[\cdot] \leftarrow \bot$
$\mathsf{T_{H_c}}[\cdot] \leftarrow \bot, \mathsf{T_{H_{ck}}}[\cdot] \leftarrow \bot, \mathsf{T_{H_{agg}}}[\cdot] \leftarrow \bot$
$\mathsf{T_b}[\cdot] \leftarrow \bot, \mathsf{T_{td}}[\cdot] \leftarrow \bot$
$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$
$H \xleftarrow{\$} \mathbb{G}$
$x \xleftarrow{\$} \mathbb{Z}_p,\ y \xleftarrow{\$} \mathbb{Z}_p \backslash \{x\}$
$X \leftarrow xG,\ Y \leftarrow yH$
$\mathsf{pk} \leftarrow (X, Y)$
$(\mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}_1^{(2)}}, \mathcal{O}_{\mathsf{Sign}_2^{(2)}}, \mathsf{H_c}, \mathsf{H_{ck}}, \mathsf{H_{agg}}}(\mathsf{par}, \mathsf{pk})$
if $[\![\mathsf{pk} \in \mathsf{vkList}^*]\!] \wedge [\![(\mathsf{M}^*) \notin \mathsf{Q_M}]\!] \wedge [\![\mathsf{T_b}[\mathsf{M}^*] = 0]\!]$ **then**
  return 0
return $\mathsf{Verify}(\mathsf{par}, \mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*)$
	$\mathsf{H_{agg}}((X, Y), \mathsf{vkList})$:
	if $[\![\mathsf{T_{H_{agg}}}[(X, Y), \mathsf{vkList}] = \bot]\!]$
	  $t \xleftarrow{\$} \mathbb{Z}_p$
	  $\mathsf{T_{H_{agg}}}[(X, Y), \mathsf{vkList}] \leftarrow t$
	$N \leftarrow \lvert\mathsf{vkList}\rvert$
	parse $((X_i, Y_i))_{i \in [N]} \leftarrow \mathsf{vkList}$
	for $i \in [N]$ **do**
	  if $[\![\mathsf{T_{H_{agg}}}[(X_i, Y_i), \mathsf{vkList}] = \bot]\!]$
	    $t_i \xleftarrow{\$} \mathbb{Z}_p$
	    $\mathsf{T_{H_{agg}}}[(X_i, Y_i), \mathsf{vkList}] \leftarrow t_i$
	return $\mathsf{T_{H_{agg}}}[(X, Y), \mathsf{vkList}]$

Figure 4.6: The fifth game Game_5 and the sixth game Game_6. The changes from the previous game are highlighted in blue.

$\epsilon_{\mathcal{A}}$ satisfies the following inequalities.

$$\begin{aligned} \epsilon_{\mathcal{A}} &= \Pr[\mathsf{Game}_1 = 1] \\ &\leq e(Q_S + 1) \Pr[\mathsf{Game}_2 = 1] \\ &\leq e(Q_S + 1)(\epsilon + \Pr[\mathsf{Game}_4 = 1]) \\ &\leq e(Q_S + 1)(2\epsilon + \Pr[\mathsf{Game}_6 = 1]) \\ &\leq e(Q_S + 1)\left(2\epsilon + \frac{2Q_H + Q_S + 2}{p}\right). \end{aligned}$$

Therefore, if $\mathbb{G}$ is a (t, ϵ)-DDH group, $\mathsf{HBMSDDH}$-1 is $(t_{\mathcal{A}}, Q_S, Q_H, N, \epsilon_{\mathcal{A}})$-2-MS-UF-2 such that

$$\begin{aligned} &\epsilon_{\mathcal{A}} \geq e(Q_S + 1)\left(2\epsilon + \frac{2Q_H + Q_S + 2}{p}\right) \text{ and} \\ &t_{\mathcal{A}} \leq \min(t_1, t_2) \text{ where} \\ &t_1 = t - (4Q_H + 6Q_S N + 4Q_S + 2N + 12)t_{\mathrm{mul}} - O(Q_H + Q_S N), \\ &t_2 = t - (2Q_H + 6Q_S N + 2Q_S + 2N + 8)t_{\mathrm{mul}} - O(Q_H + Q_S N). \end{aligned}$$

This completes the proof. □

Below, we prove Lemmas 2, 3 and 6 to 8.

Proof of Lemma 2. Here, we provide the proof of Lemma 2.

Lemma 2. $\Pr[\mathsf{Game}_1 = 1] \leq e(Q_S + 1)\Pr[\mathsf{Game}_2 = 1]$.

Proof. First, we show that the added steps of H_{ck} in Game_2 do not affect the probability of $\mathcal{A}$ winning the game. Since (U_1, U_2) in Game_2 is generated by uniformly choosing from $\mathbb{G}^2$ independently of the value of $\mathsf{T}_\mathrm{b}[\mathsf{M}]$, the distributions of the responses of H_{ck} in both games are identical. Therefore, the steps do not affect the probability of $\mathcal{A}$ winning the game.

Second, we show that $\Pr[\mathsf{Game}_1 = 1] \leq e(Q_S + 1)\Pr[\mathsf{Game}_2 = 1]$. For Game_2, let $\mathsf{E}_1^{\mathsf{Game}_2}$ be the event where the game does not terminate in the signing oracle $\mathcal{O}_{\mathsf{Sign}_2^{(2)}}$, $\mathsf{E}_2^{\mathsf{Game}_2}$ be the event where $\mathcal{A}$'s output satisfies the added condition $\mathsf{T}_\mathrm{b}[\mathsf{M}^*] = 0$, and $\mathsf{E}_3^{\mathsf{Game}_2}$ be the event where $\mathcal{A}$'s output satisfies the winning conditions as same as Game_1. Then, we have

$$\begin{aligned}\Pr[\mathsf{Game}_2 = 1] &= \Pr[\mathsf{E}_1^{\mathsf{Game}_2} \wedge \mathsf{E}_2^{\mathsf{Game}_2} \wedge \mathsf{E}_3^{\mathsf{Game}_2}] \\ &= \Pr[\mathsf{E}_1^{\mathsf{Game}_2}]\Pr[\mathsf{E}_3^{\mathsf{Game}_2}|\mathsf{E}_1^{\mathsf{Game}_2}]\Pr[\mathsf{E}_2^{\mathsf{Game}_2}|\mathsf{E}_1^{\mathsf{Game}_2} \wedge \mathsf{E}_3^{\mathsf{Game}_2}].\end{aligned}$$

Firstly, we evaluate $\Pr[\mathsf{E}_1^{\mathsf{Game}_2}]$. The game aborts in $\mathcal{O}_{\mathsf{Sign}_2^{(2)}}$ if $\mathsf{T}_\mathrm{b}[\mathsf{M}] = 0$ holds for a queried message. Thus, $\mathsf{E}_1^{\mathsf{Game}_2}$ occurs when $\mathsf{T}_\mathrm{b}[\mathsf{M}] = 1$ holds at that point for all messages queried to the signing oracle. Since the responses of the random oracles and the responses of the signing oracle $\mathcal{O}_{\mathsf{Sign}_1^{(2)}}$ leak no information on the value of $\mathsf{T}_\mathrm{b}[\mathsf{M}]$ for any m, $\mathcal{A}$ can know $\mathsf{T}_\mathrm{b}[\mathsf{M}]$ only when it observes whether the game continues or not. Also, $\mathcal{A}$ can only know $\mathsf{T}_\mathrm{b}[\mathsf{M}] = 1$ for all messages queried to $\mathcal{O}_{\mathsf{Sign}_2^{(2)}}$ as long as the game continues. Therefore, the probability that $\mathsf{T}_\mathrm{b}[\mathsf{M}] = 0$ holds for a queried message is equal to $(1 - \delta)$. In consequence, because $\mathcal{A}$ can make at most Q_S signing queries, we have $\Pr[\mathsf{E}_1^{\mathsf{Game}_2}] \geq \delta^{Q_S}$. Setting $\delta = Q_S/(Q_S + 1)$, we have $\Pr[\mathsf{E}_1^{\mathsf{Game}_2}] \geq \delta^{Q_S} \geq 1/e$. The last inequality holds because of the fact that $(1 + 1/Q_S)^{Q_S} < e$ for $Q_S > 0$.

Next, we evaluate $\Pr[\mathsf{E}_3^{\mathsf{Game}_2}|\mathsf{E}_1^{\mathsf{Game}_2}]$. Conditioned on $\mathsf{E}_1^{\mathsf{Game}_2}$, Game_2 does not terminate, and the distribution of the view of $\mathcal{A}$ in Game_2 is identical to the distribution of the view of $\mathcal{A}$ in Game_1. Thus, $\mathcal{A}$'s output in Game_2 satisfies the winning conditions of Game_1 with the same probability as in Game_1. Namely, we have $\Pr[\mathsf{E}_3^{\mathsf{Game}_2}|\mathsf{E}_1^{\mathsf{Game}_2}] = \Pr[\mathsf{Game}_1 = 1]$.

Finally, we evaluate $\Pr[\mathsf{E}_2^{\mathsf{Game}_2}|\mathsf{E}_1^{\mathsf{Game}_2} \wedge \mathsf{E}_3^{\mathsf{Game}_2}]$. Conditioned on $\mathsf{E}_1^{\mathsf{Game}_2}$ and $\mathsf{E}_3^{\mathsf{Game}_2}$, since M^* has never queried to the signing oracles, $\mathcal{A}$ cannot know $\mathsf{T}_\mathrm{b}[\mathsf{M}^*]$. Then, $\Pr[\mathsf{E}_2^{\mathsf{Game}_2}|\mathsf{E}_1^{\mathsf{Game}_2} \wedge \mathsf{E}_3^{\mathsf{Game}_2}] = (1 - \delta)$ holds. Setting $\delta = Q_S/(Q_S + 1)$, we obtain $\Pr[\mathsf{E}_2^{\mathsf{Game}_2}|\mathsf{E}_1^{\mathsf{Game}_2} \wedge \mathsf{E}_3^{\mathsf{Game}_2}] = 1/(Q_S + 1)$.

Combining all bounds, we obtain $\Pr[\mathsf{Game}_1 = 1] \leq e(Q_S + 1)\Pr[\mathsf{Game}_2 = 1]$. This completes the proof. □

Proof of Lemma 3. Here we show Lemma 3.

Lemma 3. *If $\mathbb{G}$ is a (t, ϵ)-DDH group, for $\mathcal{A}$ such that $t_{\mathcal{A}} \leq t - (4Q_H + 6Q_S N + 4Q_S + 2N + 12)t_{\mathrm{mul}} - O(Q_H + Q_S N)$, the following holds.*

$$|\Pr[\mathsf{Game}_2 = 1] - \Pr[\mathsf{Game}_3 = 1]| \leq \epsilon.$$

Proof. To prove this lemma, we construct an adversary $\mathcal{B}$ against the DDH problem that internally runs an adversary $\mathcal{A}$ against unforgeability game Game_2 or Game_3. $\mathcal{B}$ takes as input an instance of the DDH problem $((\mathbb{G}, p, G), H, P, Q)$. $\mathcal{B}$ behaves as same as the challenger in Game_2 and Game_3 except for how it generates par and (U_1, U_2) in the random oracle $\mathsf{H}_{\mathsf{ck}}(\mathsf{M})$. Specifically, it assigns $(\mathbb{G}, p, G, H)$ of input to $(\mathbb{G}, p, G, H)$ of par, instead of generating by GrGen and uniformly choosing H. Moreover, it generates (U_1, U_2) in the random oracle $\mathsf{H}_{\mathsf{ck}}(\mathsf{M})$ as follows. It first generates $(P', Q') \leftarrow \mathsf{RandDH}(G, H, P, Q)$ defined in Section 2.2.2. If $\mathsf{T}_{\mathrm{b}}[\mathsf{M}] = 0$, it assigns $(U_1, U_2) \leftarrow (P', Q')$. If $\mathsf{T}_{\mathrm{b}}[\mathsf{M}] = 1$, it computes $(U_1, U_2)^\top \leftarrow (P', Q')^\top + (X, Y)^\top$. Finally, $\mathcal{B}$ outputs 1 if $\mathcal{A}$ wins the game. Otherwise, it outputs 0.

Now we evaluate the running time $t_{\mathcal{B}}$ of $\mathcal{B}$. We assume that t_{mul} time is required for one scalar multiplication in $\mathbb{G}$, and unit time is required for the other non-cryptographic operations. $\mathcal{B}$ computes 2 scalar multiplications to generate pk. For time to answer random oracle queries, we consider only the case of H_{c} because H_{c} takes longer time than H_{ck} and $\mathsf{H}_{\mathsf{agg}}$. To respond to a query to H_{c}, $\mathcal{B}$ makes one query to H_{ck} and executes $O(1)$ other non-cryptographic operations. 4 scalar multiplications and $O(1)$ other non-cryptographic operations are required for one query to H_{ck}. Thus, in total, $\mathcal{A}$ executes 4 scalar multiplications and $O(1)$ other non-cryptographic operations to respond to one query to H_{c}. For each signing query, there are at most $6N$ scalar multiplications, one query to H_{ck}, one query to H_{c}, N queries to $\mathsf{H}_{\mathsf{agg}}$, and $O(N)$ other non-cryptographic operations, thus totally $Q_S(6N + 4)t_{\mathrm{mul}} + O(Q_S N)$ time is required for responding to all signing queries. There are $2N + 6$ scalar multiplications, one query to H_{ck}, one query to H_{c}, N queries to $\mathsf{H}_{\mathsf{agg}}$, and $O(N)$ other non-cryptographic operations to verify the adversary's output. From these evaluations and the fact that $\mathcal{A}$ runs $\mathcal{A}$ once, we obtain $t_{\mathcal{B}} \leq t_{\mathcal{A}} + (4Q_H + 6Q_S N + 4Q_S + 2N + 12)t_{\mathrm{mul}} + O(Q_H + Q_S N)$.

Now we show that $\mathsf{Adv}_{\mathcal{B}}^{\mathrm{ddh}}(1^\lambda) = |\Pr[\mathsf{Game}_2 = 1] - \Pr[\mathsf{Game}_3 = 1]|$. We can prove this equality by proving the followings.

(i): $\Pr[\mathsf{Game}_2 = 1]$ is equal to the probability that $\mathcal{B}$ outputs 1 conditioned on (G, H, P, Q) is a non-DH tuple.

(ii): $\Pr[\mathsf{Game}_3 = 1]$ is equal to the probability that $\mathcal{B}$ outputs 1 conditioned on (G, H, P, Q) is a DH tuple.

The differences between the behavior of $\mathcal{B}$ and the behavior of the challenger in Game_2 or Game_3 are the way to generate par and the way to generate (U_1, U_2) in $\mathsf{H}_{\mathsf{ck}}(\mathsf{M})$. It is clear that the first difference does not affect the probability of $\mathcal{A}$ winning the game. Therefore, to prove the above (i) and (ii), it is sufficient to prove the following (I) and (II), respectively.

(I): The distribution of the responses of $\mathsf{H}_{\mathsf{ck}}(\mathsf{M})$ in Game_2 is identical to the distribution of the responses of $\mathsf{H}_{\mathsf{ck}}(\mathsf{M})$ in $\mathcal{B}$ conditioned on (G, H, P, Q) is a non-DH tuple.

(II): The distribution of the responses of $\mathsf{H}_{\mathsf{ck}}(\mathsf{M})$ in Game_3 is identical to the distribution of the responses of $\mathsf{H}_{\mathsf{ck}}(\mathsf{M})$ in $\mathcal{B}$ conditioned on (G, H, P, Q) is a DH tuple.

Below, we prove (I) and (II).

(I): In Game_2, the challenger chooses $(U_1, U_2) \xleftarrow{\$} \mathbb{G}^2$ independently of $\mathsf{T}_{\mathrm{b}}[\mathsf{M}]$. $\mathcal{B}$ generates (P', Q') by $\mathsf{RandDH}(G, H, P, Q)$, assigns $(U_1, U_2)^\top \leftarrow (P', Q')^\top$ if $\mathsf{T}_{\mathrm{b}}[\mathsf{M}] = 0$, and computes $(U_1, U_2)^\top \leftarrow (P', Q')^\top + (X, Y)^\top$ if $\mathsf{T}_{\mathrm{b}}[\mathsf{M}] = 1$. Because of the property of RandDH, conditioned on (G, H, P, Q) is a non-DH tuple, (P', Q') is uniformly distributed over $\mathbb{G}^2$. Then, $(P', Q')^\top + (X, Y)^\top$ is also uniformly distributed over $\mathbb{G}^2$. Therefore, the responses of $\mathsf{H}_{\mathsf{ck}}(\mathsf{M})$ of $\mathcal{B}$ are uniformly distributed over $\mathbb{G}^2$ in both cases where $\mathsf{T}_{\mathrm{b}}[\mathsf{M}] = 0$ and $\mathsf{T}_{\mathrm{b}}[\mathsf{M}] = 1$. Therefore, (I) holds.

(II): In Game_3, the challenger chooses $\rho \xleftarrow{\$} \mathbb{Z}_p$, assigns $(U_1, U_2)^\top \leftarrow \rho(G, H)^\top$ if $\mathsf{T}_{\mathrm{b}}[\mathsf{M}] = 0$, and computes $(U_1, U_2)^\top \leftarrow \rho(G, H)^\top + (X, Y)^\top$ if $\mathsf{T}_{\mathrm{b}}[\mathsf{M}] = 1$. $\mathcal{B}$ generates (P', Q') by $\mathsf{RandDH}(G, H, P, Q)$, assigns $(U_1, U_2)^\top \leftarrow (P', Q')^\top$ if $\mathsf{T}_{\mathrm{b}}[\mathsf{M}] = 0$, and assigns $(U_1, U_2)^\top \leftarrow (P', Q')^\top + (X, Y)^\top$ if $\mathsf{T}_{\mathrm{b}}[\mathsf{M}] = 1$. Because of the property of RandDH, conditioned on (G, H, P, Q) is a DH tuple, (P', Q') satisfies that P' is uniformly distributed over $\mathbb{G}$ and (G, H, P', Q') is a DH tuple. Thus, the distribution of $(P', Q')^\top$ is identical to the distribution of $\rho(G, H)^\top$ where ρ is uniformly chosen from $\mathbb{Z}_p$. Therefore, (II) holds.

By combining all arguments, we obtain $\mathsf{Adv}^{\mathrm{ddh}}_{\mathcal{B}}(1^\lambda) = |\Pr[\mathsf{Game}_2 = 1] - \Pr[\mathsf{Game}_3 = 1]|$.

By assuming that $\mathbb{G}$ is a (t, ϵ)-DDH group, for $\mathcal{B}$ such that $t_{\mathcal{B}} \leq t$, we have $\mathsf{Adv}^{\mathrm{ddh}}_{\mathcal{B}}(1^\lambda) \leq \epsilon$. Since $t_{\mathcal{B}} \leq t_{\mathcal{A}} + (4Q_H + 6Q_S N + 4Q_S + 2N + 12)t_{\mathrm{mul}} + O(Q_H + Q_S N)$ and $\mathsf{Adv}^{\mathrm{ddh}}_{\mathcal{B}}(1^\lambda) = |\Pr[\mathsf{Game}_2 = 1] - \Pr[\mathsf{Game}_3 = 1]|$, if $\mathbb{G}$ is a (t, ϵ)-DDH group, for $\mathcal{A}$ such that $t_{\mathcal{A}} \leq t - (4Q_H + 6Q_S N + 4Q_S + 2N + 12)t_{\mathrm{mul}} - O(Q_H + Q_S N)$, the following inequality holds.

$$|\Pr[\mathsf{Game}_2 = 1] - \Pr[\mathsf{Game}_3 = 1]| \leq \epsilon.$$

This completes the proof. □

Proof of Lemma 6. Here we prove Lemma 6. To prove Lemma 6, we use the following lemma.

Lemma 4. *Let* $\mathsf{Game}^{\mathrm{eqv}}(1^\lambda)$ *be the following game between a challenger and an adversary* $\mathcal{A}$*, which is also depicted in Fig. 4.7.*

Setup: *The challenger generates* $(\mathbb{G}, p, G)$ *by* GrGen. *It sends* $(\mathbb{G}, p, G)$ *to* $\mathcal{A}$ *and receives* $H \in \mathbb{G}$ *and* $x \in \mathbb{Z}_p$ *from* $\mathcal{A}$. *It computes* $(X, Y)^\top \leftarrow x(G, H)^\top$ *and initializes a table* $\mathsf{T}_S[\cdot]$. *It chooses a bit* $b \xleftarrow{\$} \{0, 1\}$.

Oracles: *The challenger allows* $\mathcal{A}$ *to access to the following oracles concurrently at most* Q *times. Note that* $\mathcal{A}$ *is allowed to make only one query for each session identifier* I*, which is included in each query to oracles.*

$\mathcal{O}_{\mathrm{eqv1}}(b, \cdot, \cdot)$**:** *As a query, the challenger receives a session identifier* I *and* $\rho \in \mathbb{Z}_p$. *It computes* $(U_1, U_2)^\top \leftarrow \rho(G, H)^\top + (X, Y)^\top$. *It responds as follows.*

Case $b = 0$**:** *It chooses* $r, d \xleftarrow{\$} \mathbb{Z}_p$ *and computes* $T \leftarrow d(U_1, U_2)^\top + r(G, H)^\top$. *It stores* $\mathsf{T}_S[I] \leftarrow ((U_1, U_2), \rho, r, d, T)$ *and returns* T.

Case $b = 1$**:** *It chooses* $\alpha, \beta \xleftarrow{\$} \mathbb{Z}_p$ *and computes* $T \leftarrow \alpha(G, H)^\top + \beta(X, Y)$. *It stores* $\mathsf{T}_S[I] \leftarrow ((U_1, U_2), \rho, \alpha, \beta, T)$ *and returns* T.

$\mathcal{O}_{\mathrm{eqv2}}(b, \cdot, \cdot)$**:** *As a query, the challenger receives a session identifier* I *and* $c \in \mathbb{Z}_p$. *If* $\mathsf{T}_S[I]$ *is empty, then it return* $\perp$. *Otherwise, it responds as follows.*

Case $b = 0$**:** *The challenger looks up* $((U_1, U_2), \rho, r, d, T)$ *from* $\mathsf{T}_S[I]$, *computes* $s \leftarrow xc + r \mod p$ *and returns* (d, s).

Case $b = 1$**:** *The challenger looks up* $((U_1, U_2), \rho, \alpha, \beta, T)$ *from* $\mathsf{T}_S[I]$, *computes* $d \leftarrow \beta + c \mod p$ *and* $s \leftarrow \alpha - d\rho \mod p$ *and returns* (d, s).

Guess: *Finally,* $\mathcal{A}$ *outputs a guess* $b' \in \{0, 1\}$. *If* $b = b'$ *holds, then* $\mathcal{A}$ *wins this game.*

The advantage of $\mathcal{A}$ *against the above game is defined as*

$$\mathsf{Adv}^{\mathrm{eqv}}_{\mathcal{A}}(1^\lambda) = \left|\Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0]\right|.$$

For any computationally unbounded adversary $\mathcal{A}$, $\mathsf{Adv}^{\mathrm{eqv}}_{\mathcal{A}}(1^\lambda) = 0$ *holds.*

$\mathsf{Game}^{\mathrm{eqv}}(1^\lambda)$

$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$
$(H, x, \mathsf{st}_{\mathcal{A}}) \xleftarrow{\$} \mathcal{A}(\mathbb{G}, p, G)$ // $H \in \mathbb{G}, x \in \mathbb{Z}_p$
$(X, Y)^\top \leftarrow x(G, H)^\top$
$\mathsf{T}_S[\cdot] \leftarrow \bot$
$b \xleftarrow{\$} \{0, 1\}$
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathrm{eqv1}}, \mathcal{O}_{\mathrm{eqv2}}}(\mathsf{st}_{\mathcal{A}})$
return $(b = b')$

$\mathcal{O}_{\mathrm{eqv1}}(b, I, \rho)$

req $[\![\rho \in \mathbb{Z}_p]\!]$
$(U_1, U_2)^\top \leftarrow \rho(G, H)^\top + (X, Y)^\top$
if $b = 0$ **then**
  $r, d \xleftarrow{\$} \mathbb{Z}_p$
  $T \leftarrow d(U_1, U_2)^\top + r(G, H)^\top$
  $\mathsf{T}_S[I] \leftarrow ((U_1, U_2), \rho, r, d, T)$
else
  $\alpha, \beta \xleftarrow{\$} \mathbb{Z}_p$
  $T \leftarrow \alpha(G, H)^\top + \beta(X, Y)$
  $\mathsf{T}_S[I] \leftarrow ((U_1, U_2), \rho, \alpha, \beta, T)$
return T

$\mathcal{O}_{\mathrm{eqv2}}(b, I, c)$

req $[\![\mathsf{T}_S[I] \neq \bot]\!] \wedge [\![c \in \mathbb{Z}_p]\!]$
if $b = 0$ **then**
  $((U_1, U_2), \rho, r, d, T) \leftarrow \mathsf{T}_S[I]$
  $s \leftarrow xc + r \mod p$
else
  $((U_1, U_2), \rho, \alpha, \beta, T) \leftarrow \mathsf{T}_S[I]$
  $d \leftarrow \beta + c \mod p$
  $s \leftarrow \alpha - d\rho \mod p$
return (d, s)

Figure 4.7: The game $\mathsf{Game}^{\mathrm{eqv}}$.

Before we prove Lemma 4, we explain the intuition of the proof.

To prove this lemma, we should prove that, in $\mathsf{Game}^{\mathrm{eqv}}$, the statistical distance between the distribution of an adversary's view in the case where $b = 0$ and the distribution of an adversary's view in the case where $b = 1$ is equal to 0. However, it is hard to prove it directly because an adversary can *concurrently* access *stateful* oracles.

To overcome this difficulty, we prove Lemma 4 step by step. We resolve the difficulty arising from concurrently accessing by using the hybrid argument. To carry out this strategy, we consider the intermediate game $\mathsf{Game}_k^{\mathrm{eqv}}$ in which an adversary is allowed to access the *stateful* oracles that switch behavior on the k-th query. Moreover, to evaluate the advantage of an adversary in this game, we consider the simple game $\mathsf{Game}_0^{\text{eqv}}$ in which an adversary needs to make all queries to the interactive oracles at the beginning of the game. We prove the advantage of an adversary in $\mathsf{Game}_0^{\text{eqv}}$ is 0 (in Lemma 5), and by using this, we prove the advantage of an adversary in $\mathsf{Game}_k^{\text{eqv}}$ is also 0.

Now we start the proof of Lemma 4. First, we prove the following lemma.

Lemma 5. *We consider the following game $\mathsf{Game}_0^{\text{eqv}}$ between a challenger and an adversary $\mathcal{A}$, which is depicted in Fig. 4.8.*

Setup: *The challenger generates $(\mathbb{G}, p, G)$ by GrGen. It sends $(\mathbb{G}, p, G)$ to $\mathcal{A}$ and receives $H \in \mathbb{G}$ and $x, \rho, c \in \mathbb{Z}_p$ from $\mathcal{A}$. It computes $(X, Y)^\top \leftarrow x(G, H)^\top$ and $(U_1, U_2)^\top \leftarrow \rho(G, H)^\top + (X, Y)^\top$. It chooses a bit $b \xleftarrow{\$} \{0, 1\}$. It allows $\mathcal{A}$ to access to the following oracle only once.*

Oracle $\mathcal{O}_{\text{eqv0}}(b)$: *The challenger responds as follows.*

Case $b = 0$: *The challenger chooses $r, d \xleftarrow{\$} \mathbb{Z}_p$, computes $T \leftarrow d(U_1, U_2)^\top + r(G, H)^\top$ and $s \leftarrow xc + r \mod p$ and returns (T, d, s).*

Case $b = 1$: *The challenger chooses $\alpha, \beta \xleftarrow{\$} \mathbb{Z}_p$ and computes $T \leftarrow \alpha(G, H)^\top + \beta(X, Y)$, $d \leftarrow \beta + c \mod p$, and $s \leftarrow \alpha - d\rho \mod p$ and returns (T, d, s).*

Guess: *Finally, $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$. If $b = b'$ holds, then $\mathcal{A}$ wins this game.*

The advantage of $\mathcal{A}$ against the above game is defined as

$$\mathsf{Adv}_{\mathcal{A}}^{\text{eqv0}}(1^\lambda) = |\Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0]| \,.$$

For any computationally unbounded adversary $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\text{eqv0}}(1^\lambda) = 0$ holds.

Proof. For $W \in \{V \in \mathbb{G}^2 | V = v(G, H)^\top, v \in \mathbb{Z}_p\}$, let $\log_{(G,H)} W$ be the element $w \in \mathbb{Z}_p$ s.t. $W = w(G, H)^\top$. Below, we write $\mathcal{O}_{\text{eqv0}}$'s response (T, z, s) using matrices and vectors with $\mathbb{Z}_p$ coefficients.

- In the case $b = 0$, the response of $\mathcal{O}_{\text{eqv0}}$ satisfies $T = d(U_1, U_2)^\top + r(G, H)^\top = (r + (\rho + x)d)(G, H)^\top$ and $s = xc + r \mod p$ where $r, d \xleftarrow{\$} \mathbb{Z}_p$. Thus, we obtain

$$\begin{pmatrix} \log_{(G,H)} T \\ d \\ s \end{pmatrix} = \begin{pmatrix} 1 & \rho + x \\ 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} r \\ d \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ xc \end{pmatrix}. \tag{4.1}$$

$\mathsf{Game}_0^{\mathrm{eqv}}(1^\lambda)$	$\mathcal{O}_{\mathrm{eqv0}}(b)$
$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$	**if** $b = 0$ **then**
$(H, x, \rho, c, \mathsf{st}_{\mathcal{A}}) \xleftarrow{\$} \mathcal{A}(\mathbb{G}, p, G)$ // $H \in \mathbb{G}, x, \rho, c \in \mathbb{Z}_p$	$\quad r, d \xleftarrow{\$} \mathbb{Z}_p$
$(X, Y)^\top \leftarrow x(G, H)^\top$	$\quad T \leftarrow d(U_1, U_2)^\top + r(G, H)^\top$
$(U_1, U_2)^\top \leftarrow \rho(G, H)^\top + (X, Y)^\top$	$\quad s \leftarrow xc + r \mod p$
$b \xleftarrow{\$} \{0, 1\}$	**else**
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathrm{eqv1}}, \mathcal{O}_{\mathrm{eqv2}}}(\mathsf{st}_{\mathcal{A}})$	$\quad \alpha, \beta \xleftarrow{\$} \mathbb{Z}_p$
return $(b = b')$	$\quad T \leftarrow \alpha(G, H)^\top + \beta(X, Y)$
	$\quad d \leftarrow \beta + c \mod p$
	$\quad s \leftarrow \alpha - d\rho \mod p$
	return (T, d, s)

Figure 4.8: The game $\mathsf{Game}_0^{\mathrm{eqv}}$.

- In the case $b = 1$, the response of $\mathcal{O}_{\mathrm{eqv0}}$ satisfies $T = \alpha(G, H)^\top + \beta(X, Y)^\top = (\alpha + \beta x)(G, H)^\top$, $d = \beta + c \mod p$, and $s = \alpha - d\rho \mod p$ where $\alpha, \beta \xleftarrow{\$} \mathbb{Z}_p$. Thus, we obtain

$$\begin{pmatrix} \log_{(G,H)} T \\ d \\ s \end{pmatrix} = \begin{pmatrix} 1 & x \\ 0 & 1 \\ 1 & -\rho \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} + \begin{pmatrix} 0 \\ c \\ -c\rho \end{pmatrix}. \tag{4.2}$$

The advantage of $\mathcal{A}$ in $\mathsf{Game}_0^{\mathrm{eqv}}$ is equal to the statistical distance between the distribution of the response of $\mathcal{O}_{\mathrm{eqv0}}$ in the case $b = 0$ and that in the case $b = 1$. Therefore, to prove $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{eqv0}}(1^\lambda) = 0$ for any computationally unbounded adversary $\mathcal{A}$, we prove that the distribution of $(\log_{(G,H)} T, d, s)^\top$ in Eq. (4.1) is identical to that in Eq. (4.2) when $r, d \xleftarrow{\$} \mathbb{Z}_p$ and $\alpha, \beta \xleftarrow{\$} \mathbb{Z}_p$.

For a matrix C, let $\mathrm{Im}(C)$ denote the column space of C. Let D_0 and D_1 be the column spaces as follows.

$$D_0 = \mathrm{Im} \begin{pmatrix} 1 & \rho + x \\ 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad D_1 = \mathrm{Im} \begin{pmatrix} 1 & x \\ 0 & 1 \\ 1 & -\rho \end{pmatrix}.$$

Note that the distribution of $(\log_{(G,H)} T, d, s)^\top$ in Eq. (4.1) and that in Eq. (4.2) are identical if and only if

$$D_0 + (0, 0, xc)^\top = D_1 + (0, c, -c\rho)^\top$$

holds, where the above equality means the equality of the left and the right affine subspaces. Furthermore, the above equality holds when the followings hold.

- $D_0 = D_1$.
- $(0, 0, xc)^\top - (0, c, -c\rho)^\top \in D_0$.

Now, we prove $D_0 = D_1$ by showing $D_0 \subseteq D_1$ and $D_1 \subseteq D_0$. For any $z_0 \in D_0$, we can write z_0 as follows:

$$\begin{aligned} z_0 &= r \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + d \begin{pmatrix} \rho + x \\ 1 \\ 0 \end{pmatrix} \\ &= r \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + d \begin{pmatrix} \rho \\ 0 \\ \rho \end{pmatrix} - d \begin{pmatrix} \rho \\ 0 \\ \rho \end{pmatrix} + d \begin{pmatrix} \rho + x \\ 1 \\ 0 \end{pmatrix} \\ &= (r + d\rho) \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + d \begin{pmatrix} x \\ 1 \\ -\rho \end{pmatrix} \in D_1. \end{aligned}$$

where $r, d \in \mathbb{Z}_p$. Thus, any $z_0 \in D_0$ is in D_1. This implies $D_0 \subseteq D_1$. On the other hand, for any $z_1 \in D_1$, we can write z_1 as follows:

$$\begin{aligned} z_1 &= \alpha \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + \beta \begin{pmatrix} x \\ 1 \\ -\rho \end{pmatrix} \\ &= \alpha \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} - \beta \begin{pmatrix} \rho \\ 0 \\ \rho \end{pmatrix} + \beta \begin{pmatrix} \rho \\ 0 \\ \rho \end{pmatrix} + \beta \begin{pmatrix} x \\ 1 \\ -\rho \end{pmatrix} \\ &= (\alpha - \beta\rho) \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + \beta \begin{pmatrix} \rho + x \\ 1 \\ 0 \end{pmatrix} \in D_0. \end{aligned}$$

where $\alpha, \beta \in \mathbb{Z}_p$. Thus, any $z_1 \in D_1$ is in D_0. This implies $D_1 \subseteq D_0$.

Next, we show $(0, 0, xc)^\top - (0, c, -c\rho)^\top \in D_0$. This holds because, for $(0, 0, xc)^\top$ and $(0, c, -c\rho)^\top$, we have

$$\begin{pmatrix} 0 \\ 0 \\ xc \end{pmatrix} - \begin{pmatrix} 0 \\ c \\ -c\rho \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ xc \end{pmatrix} + \begin{pmatrix} c(\rho + x) \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} c(\rho + x) \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ c \\ -c\rho \end{pmatrix}$$

$$= c(x+\rho)\begin{pmatrix}1\\0\\1\end{pmatrix} - c\begin{pmatrix}\rho+x\\1\\0\end{pmatrix} \in D_0.$$

This completes the proof. □

Now we show Lemma 4 from the above lemma.

Proof of Lemma 4. We consider the following game $\mathsf{Game}_k^{\mathrm{eqv}}(1^\lambda)$ where $k \in [Q]$ between a challenger and an adversary, which is depicted in Fig. 4.9.

Setup: The challenger generates $(\mathbb{G}, p, G)$ by GrGen. It sends $(\mathbb{G}, p, G)$ to $\mathcal{A}$ and receives $H \in \mathbb{G}$ and $x \in \mathbb{Z}_p$ from $\mathcal{A}$. It computes $(X, Y)^\top \leftarrow x(G, H)^\top$. It initializes tables $\mathsf{T}_S[\cdot]$ and a counter $\mathsf{ctr} = 1$. It chooses a bit $b \xleftarrow{\$} \{0, 1\}$.

Oracles: The challenger allows $\mathcal{A}$ to access to the following oracles concurrently at most Q times. Note that $\mathcal{A}$ is allowed to make only one query for each session identifier I, which is included in each query to oracles. We assume that the adversary sequentially generates I from 1 to Q.

$\mathcal{O}_{\mathrm{eqv1},k}(b, \cdot, \cdot)$**:** As a query, the challenger receives a session identifier I and $\rho \in \mathbb{Z}_p$. It computes $(U_1, U_2)^\top \leftarrow \rho(G, H)^\top + (X, Y)^\top$. It responds as follows.

Case $(I > k)$ **or** $(I = k) \wedge (b = 0)$**:** The challenger responds as $\mathcal{O}_{\mathrm{eqv1}}(0, \cdot, \cdot)$ in $\mathsf{Game}^{\mathrm{eqv}}$.

Case $(I < k)$ **or** $(I = k) \wedge (b = 1)$**:** The challenger responds as $\mathcal{O}_{\mathrm{eqv1}}(1, \cdot, \cdot)$ in $\mathsf{Game}^{\mathrm{eqv}}$.

$\mathcal{O}_{\mathrm{eqv2},k}(b, \cdot, \cdot)$**:** The challenger receives a session identifier I and $c \in \mathbb{Z}_p$ as a query. If $\mathsf{T}_S[I]$ is empty, then it return $\perp$. Otherwise, it responds as follows.

Case $I > k$ **or** $(I = k) \wedge (b = 0)$**:** The challenger responds as $\mathcal{O}_{\mathrm{eqv2}}(0, \cdot, \cdot)$ in $\mathsf{Game}^{\mathrm{eqv}}$.

Case $I < k$ **or** $(I = k) \wedge (b = 1)$**:** The challenger responds as $\mathcal{O}_{\mathrm{eqv2}}(1, \cdot, \cdot)$ in $\mathsf{Game}^{\mathrm{eqv}}$.

Guess: Finally, $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$. If $b = b'$ holds, then $\mathcal{A}$ wins this game.

Then, the advantage of $\mathcal{A}$ against the above game is defined as

$$\mathsf{Adv}_{\mathcal{A}}^{\mathrm{eqv},k}(1^\lambda) = \left|\Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0]\right|.$$

$\mathsf{Game}_k^{\mathrm{eqv}}(1^\lambda)$

$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$
$(H, x, \mathsf{st}_{\mathcal{A}}) \xleftarrow{\$} \mathcal{A}(\mathbb{G}, p, G)$ // $H \in \mathbb{G}, x \in \mathbb{Z}_p$
$(X, Y)^\top \leftarrow x(G, H)^\top$
$\mathsf{T}_S[\cdot] \leftarrow \bot$
$b \xleftarrow{\$} \{0, 1\}$
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathrm{eqv1},k}, \mathcal{O}_{\mathrm{eqv2},k}}(\mathsf{st}_{\mathcal{A}})$
return $(b = b')$

$\mathcal{O}_{\mathrm{eqv1},k}(b, I, \rho)$

req $[\![\rho \in \mathbb{Z}_p]\!]$
$(U_1, U_2)^\top \leftarrow \rho(G, H)^\top + (X, Y)^\top$
if $[\![I > k]\!] \vee [\![(I = k \wedge b = 0)]\!]$ **then**
  $r, d \xleftarrow{\$} \mathbb{Z}_p$
  $T \leftarrow d(U_1, U_2)^\top + r(G, H)^\top$
  $\mathsf{T}_S[I] \leftarrow ((U_1, U_2), \rho, r, d, T)$
else
  $\alpha, \beta \xleftarrow{\$} \mathbb{Z}_p$
  $T \leftarrow \alpha(G, H)^\top + \beta(X, Y)$
  $\mathsf{T}_S[I] \leftarrow ((U_1, U_2), \rho, \alpha, \beta, T)$
return T

$\mathcal{O}_{\mathrm{eqv2},k}(b, I, c)$

req $[\![\mathsf{T}_S[I] \neq \bot]\!] \wedge [\![c \in \mathbb{Z}_p]\!]$
if $[\![I > k]\!] \vee [\![(I = k \wedge b = 0)]\!]$ **then**
  $((U_1, U_2), \rho, r, d, T) \leftarrow \mathsf{T}_S[I]$
  $s \leftarrow xc + r \mod p$
else
  $((U_1, U_2), \rho, \alpha, \beta, T) \leftarrow \mathsf{T}_S[I]$
  $d \leftarrow \beta + c \mod p$
  $s \leftarrow \alpha - d\rho \mod p$
return (d, s)

Figure 4.9: The game $\mathsf{Game}_k^{\mathrm{eqv}}$.

Bellow, we show that $\mathsf{Adv}_{\mathcal{A}'}^{\mathrm{eqv},k}(1^\lambda) = 0$ for any computationally unbounded adversary $\mathcal{A}'$ and any $k \in [Q]$ from Lemma 5. To show this, we construct an adversary $\mathcal{B}$ against the game $\mathsf{Game}_0^{\mathrm{eqv}}$ in Lemma 5 from $\mathcal{A}'$ as follows. $\mathcal{B}$ takes as inputs $(\mathbb{G}, p, G)$. It first sends it to $\mathcal{A}'$ and receives (H, x) from $\mathcal{A}'$. It computes $(X, Y)^\top \leftarrow x(G, H)^\top$ and initializes a table $\mathsf{T}_S[\cdot]$. $\mathcal{B}$ responds the queries to oracles as same as them in $\mathsf{Game}_k^{\mathrm{eqv}}$ when $I \neq k$. When $I = k$, it responds the query by accessing the oracle $\mathcal{O}_{\mathrm{eqv}0}$ in $\mathsf{Game}_0^{\mathrm{eqv}}$. Specifically, in $\mathcal{O}_{\mathrm{eqv1},k}$, it uniformly chooses $c' \xleftarrow{\$} \mathbb{Z}_p$ and outputs (H, x, ρ, c') with $\mathsf{st}_{\mathcal{A}'}$, that is stored all state information of $\mathcal{A}'$. Then, it takes as input $\mathsf{st}_{\mathcal{A}'}$, obtains (T, d, s) by accessing to $\mathcal{O}_{\mathrm{eqv}0}$, stores $\mathsf{T}_S[I] \leftarrow (d, s, c')$ and returns T to $\mathcal{A}'$. In $\mathcal{O}_{\mathrm{eqv2},k}$, it halts with output 0 if $c \neq c'$ where c is queried from $\mathcal{A}'$. Otherwise, it returns (d, s). Eventually, it obtains a guess b' from $\mathcal{A}'$ and returns b'.

Now we evaluate the relation between advantages of $\mathcal{B}$ and $\mathcal{A}'$. $\mathcal{B}$ outputs the guess b' of $\mathcal{A}'$ if it does not halt because of $c = c'$ in $\mathcal{O}_{\mathrm{eqv2},k}$ in the case $I = k$. So, we get the following equation.

$$\mathsf{Adv}_{\mathcal{B}}^{\mathrm{eqv0}}(1^\lambda) = |\Pr[b' = 1 \wedge c = c'|b = 1] - \Pr[b' = 1 \wedge c = c'|b = 0]| \quad (4.3)$$

where b is a bit chosen by the challenger in $\mathsf{Game}_0^{\mathrm{eqv}}$. Since $\mathcal{O}_{\mathrm{eqv0}}$ generates T without using c' in both cases where $b = 0$ and $b = 1$, $\mathcal{A}'$ obtain no information about c' before $\mathcal{A}'$ makes a query (k, c) to $\mathcal{O}_{\mathrm{eqv2},k}$. Also, c' is uniformly chosen from $\mathbb{Z}_p$. Therefore, we have $\Pr[c = c'|b = 1] = \Pr[c = c'|b = 0] = 1/p$. Then, we obtain

$$\begin{aligned} &|\Pr[b' = 1 \wedge c = c'|b = 1] - \Pr[b' = 1 \wedge c = c'|b = 0]| \\ &= \frac{1}{p} |\Pr[b' = 1|c = c' \wedge b = 1] - \Pr[b' = 1|c = c' \wedge b = 0]| . \end{aligned} \quad (4.4)$$

In the k-th query to $\mathcal{O}_{\mathrm{eqv1},k}$, conditioned on $c = c'$, $\mathcal{O}_{\mathrm{eqv0}}$ generates (T, d, s) in the same way to $\mathcal{O}_{\mathrm{eqv1}}$ and $\mathcal{O}_{\mathrm{eqv2}}$. Thus, for all $b \in \{0, 1\}$, conditioned on $c = c'$, the distribution of the responses of $\mathcal{O}_{\mathrm{eqv1},k}$ and $\mathcal{O}_{\mathrm{eqv2},k}$ in $\mathcal{B}$ is identical to the distribution of the responses of them in the real game $\mathsf{Game}_k^{\mathrm{eqv}}$, respectively. Then, from Eqs. (4.3) and (4.4), we have

$$\mathsf{Adv}_{\mathcal{B}}^{\mathrm{eqv0}}(1^\lambda) = \frac{1}{p}\mathsf{Adv}_{\mathcal{A}'}^{\mathrm{eqv},k}(1^\lambda). \quad (4.5)$$

From Lemma 5, $\mathsf{Adv}_{\mathcal{B}}^{\mathrm{eqv0}}(1^\lambda) = 0$ holds. Therefore, for any computationally unbounded adversary $\mathcal{A}'$ and any $k \in [Q]$, we have $\mathsf{Adv}_{\mathcal{A}'}^{\mathrm{eqv},k}(1^\lambda) = 0$.

From here, we evaluate $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{eqv}}(1^\lambda)$, which is the advantage of $\mathcal{A}$ in $\mathsf{Game}^{\mathrm{eqv}}$ for any computationally unbounded adversary $\mathcal{A}$. By the hybrid argument, we have

$$\mathsf{Adv}_{\mathcal{A}}^{\mathrm{eqv}}(1^\lambda) \leq \sum_{i=1}^{Q} \mathsf{Adv}_{\mathcal{A}'}^{\mathrm{eqv},k}(1^\lambda).$$

Since $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{eqv},k}(1^\lambda) = 0$ for all $k \in [Q]$, we have $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{eqv}}(1^\lambda) \leq 0$. Also $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{eqv}}(1^\lambda) \geq 0$ because the advantage is a non-negative real number. Therefore, we obtain $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{eqv}}(1^\lambda) = 0$. This completes the proof. □

Now we prove Lemma 6 by using this lemma.

Lemma 6. $\Pr[\mathsf{Game}_3 = 1] = \Pr[\mathsf{Game}_4 = 1]$.

Proof. To show this lemma, we construct an adversary $\mathcal{B}$ against the game $\mathsf{Game}^{\mathrm{eqv}}$ in Lemma 4, where $Q = NQ_S$, from an adversary $\mathcal{A}$ against the unforgeability game Game_3 or Game_4. $\mathcal{B}$ takes as inputs $(\mathbb{G}, p, G)$. It chooses $H \xleftarrow{\$} \mathbb{G}$ and assigns $(\mathbb{G}, q, G, H)$ to $(\mathbb{G}, q, G, H)$ of a public parameter par. It additionally initializes a counter ctr to count the number of times of oracle accessing. It executes the remaining part of the setup as same as in Game_3 or Game_4. After that, it outputs (H, sk) with a state $\mathsf{st}_{\mathcal{B}}$ and receives $\mathsf{st}_{\mathcal{B}}$ from the challenger in $\mathsf{Game}^{\mathrm{eqv}}$. Then, it runs $\mathcal{A}$ on inputs par and pk. For the random oracle queries, it responds them as in Game_3 or Game_4. In the signing oracles, while it behaves as same as those in Game_3 or Game_4 when $\mathsf{T}_{\mathrm{b}}[\mathsf{M}] = 0$, it produces the responses by accessing the oracles $\mathcal{O}_{\mathrm{eqv1}}$ and $\mathcal{O}_{\mathrm{eqv2}}$ in $\mathsf{Game}^{\mathrm{eqv}}$ when $\mathsf{T}_{\mathrm{b}}[\mathsf{M}] = 1$. Specifically, in $\mathcal{O}_{\mathsf{Sign}_1^{(2)}}$, for all $i \in \mathsf{HS}$, it sets $I_i \leftarrow \mathsf{ctr}$, obtains T_i by querying (I_i, ρ) to $\mathcal{O}_{\mathrm{eqv1}}$, computes $\mathsf{ctr} \leftarrow \mathsf{ctr} + 1$ and sets $\mathsf{st}_i \leftarrow (t_i, T_i, I_i, \widetilde{\mathsf{pk}})$. After that, it stores $\mathsf{Q}_{\mathsf{st}}[\mathsf{sid}] \leftarrow (\mathsf{vkList}, \mathsf{M}, \mathsf{HS}, (\mathsf{st}_i)_{i \in \mathsf{HS}})$ and returns $(T_i)_{i \in \mathsf{HS}}$. In $\mathcal{O}_{\mathsf{Sign}_2^{(2)}}$, it looks up $(\mathsf{vkList}, \mathsf{M}, \mathsf{HS}, (\mathsf{st}_i)_{i \in \mathsf{HS}})$ from $\mathsf{Q}_{\mathsf{st}}[\mathsf{sid}]$ and $(t_i, T_i, I_i, \widetilde{\mathsf{pk}}) \leftarrow \mathsf{st}_i$ for all $i \in \mathsf{HS}$, computes $\widetilde{T}$ and c as same as the signing oracle in Game_3 or Game_4. Then, for all $i \in \mathsf{HS}$, it obtains (s_i, d_i) by querying $(I_i, t_i c)$ to $\mathcal{O}_{\mathrm{eqv2}}$ and returns $\{(d_i, s_i)\}_{i \in \mathsf{HS}}$. Eventually, $\mathcal{B}$ returns 1 if $\mathcal{A}$ wins the game. Otherwise, it returns 0.

Now, we evaluate the advantage of $\mathcal{B}$. For the signing oracle simulated by $\mathcal{B}$, the distribution of responses is the same as in Game_3 when $b = 0$ where b is a bit chosen by the challenger in $\mathsf{Game}^{\mathrm{eqv}}$. That also is the same as that in Game_4 when $b = 1$. Thus, we have

$$\begin{aligned}\mathsf{Adv}_{\mathcal{B}}^{\mathrm{eqv}}(1^\lambda) &= |\Pr[b' = 1 | b = 0] - \Pr[b' = 1 | b = 1]| \\ &= |\Pr[\mathsf{Game}_3 = 1] - \Pr[\mathsf{Game}_4 = 1]|\end{aligned}$$

where b' is $\mathcal{B}$'s output. From Lemma 4, we have $\mathsf{Adv}_{\mathcal{B}}^{\mathrm{eqv}}(1^\lambda) = 0$. Thus, we obtain $\Pr[\mathsf{Game}_3 = 1] = \Pr[\mathsf{Game}_4 = 1]$. This completes the proof. □

Proof of Lemma 7. Here we provide the proof of Lemma 7.

Lemma 7. *If $\mathbb{G}$ is a (t, ϵ)-DDH group, for $\mathcal{A}$ such that $t_{\mathcal{A}} \leq t - (2Q_H + 6Q_S N + 2Q_S + 2N + 8)t_{\mathrm{mul}} - O(Q_H + Q_S N)$, the following inequality holds.*

$$|\Pr[\mathsf{Game}_4 = 1] - \Pr[\mathsf{Game}_5 = 1]| \leq \epsilon.$$

Proof. We construct an adversary $\mathcal{B}$ against the DDH problem that internally runs $\mathcal{A}$. $\mathcal{B}$ takes as inputs a DDH problem instance $(G, H, Y, Z) \in \mathbb{G}^4$. It behaves as same as the challenger in Game_4 or Game_5 except for the setup phase. Specifically, it uses H instead of uniformly choosing H from $\mathbb{G}$ for a public parameter par and assigns $(\mathsf{pk}, \mathsf{sk}) \leftarrow ((X, Y), \bot)$. Eventually, it outputs 1 if $\mathcal{A}$ wins the game. Otherwise, it returns 0.

Now we consider the running time $t_{\mathcal{B}}$ of $\mathcal{B}$. For the signing queries, responding a query to $\mathcal{O}_{\mathsf{Sign}_1^{(2)}}$ and a query to $\mathcal{O}_{\mathsf{Sign}_2^{(2)}}$ requires at most $6N$ scalar multiplications, $O(N)$ other non-cryptographic operations, a query to H_{c}, a query to H_{ck}, and $O(N)$ queries to $\mathsf{H}_{\mathsf{agg}}$. For random oracle queries, we only evaluate the cost of H_{ck} because H_{ck} is more expensive than H_{c} and $\mathsf{H}_{\mathsf{agg}}$. In one query to H_{ck}, there are 2 scalar multiplications and $O(1)$ other non-cryptographic operations. To verify the output of $\mathcal{A}$, it computes $2N+6$ scalar multiplications and $O(N)$ other non-cryptographic operations, and makes a query to H_{c}, a query to H_{ck}, and $O(N)$ queries to $\mathsf{H}_{\mathsf{agg}}$. From these evaluations and the fact that $\mathcal{B}$ runs $\mathcal{A}$ once, we obtain $t_{\mathcal{B}} \leq t_{\mathcal{A}} + (2Q_H + 6Q_S N + 2Q_S + 2N + 8)t_{\mathrm{mul}} + O(Q_H + Q_S N)$.

Below, we evaluate the advantage of $\mathcal{B}$. $\mathcal{B}$ can respond to the signing oracles though $\mathcal{B}$ does not have the secret key because the secret key no longer be used in the signing oracle due to the modification we made in Game_4. If (G, H, Y, Z) is a DH tuple, the distribution of pk is the same as that in Game_4. If (G, H, Y, Z) is a non-DH tuple, the distribution of pk is the same as that in Game_5. Then, we have

$$\begin{aligned}
&\mathsf{Adv}_{\mathcal{B}}^{\mathrm{ddh}}(1^\lambda) \\
&= |\Pr[b' = 1 | (G, H, P, Q) \text{ is a DH tuple}] \\
&\quad - \Pr[b' = 1 | (G, H, P, Q) \text{ is a non-DH tuple}]| \\
&= |\Pr[\mathsf{Game}_4 = 1] - \Pr[\mathsf{Game}_5 = 1]|
\end{aligned}$$

where b' is $\mathcal{B}$'s output.

By assuming that $\mathbb{G}$ is a (t, ϵ)-DDH group, for $\mathcal{B}$ such that $t_{\mathcal{B}} \leq t$, we have $\mathsf{Adv}_{\mathcal{B}}^{\mathrm{ddh}}(1^\lambda) \leq \epsilon$. Since $t_{\mathcal{B}} \leq t_{\mathcal{A}} + (2Q_H + 6Q_S N + 2Q_S + 2N + 8)t_{\mathrm{mul}} + O(Q_H + Q_S N)$ and $\mathsf{Adv}_{\mathcal{B}}^{\mathrm{ddh}}(1^\lambda) = |\Pr[\mathsf{Game}_4 = 1] - \Pr[\mathsf{Game}_5 = 1]|$, if $\mathbb{G}$ is a (t, ϵ)-DDH group, for $\mathcal{A}$ such that $t_{\mathcal{A}} \leq t - (2Q_H + 6Q_S N + 2Q_S + 2N + 8)t_{\mathrm{mul}} - O(Q_H + Q_S N)$, the following inequality holds.

$$|\Pr[\mathsf{Game}_4 = 1] - \Pr[\mathsf{Game}_5 = 1]| \leq \epsilon.$$

This completes the proof. □

Proof of Lemma 8. Here we show Lemma 8.

Lemma 8.

$$\Pr[\mathsf{Game}_6 = 1] \leq \frac{2Q_H + Q_S + 2}{p}.$$

Proof. At the end of Game_6, $\mathcal{A}$ outputs $\mathsf{M}^*, \mathsf{vkList}^*$, and $\widetilde{\mathsf{sig}}^* = (c^*, \tilde{d}^*, \tilde{s}^*)$. If $\mathcal{A}$ wins, then $(X, Y) \in \mathsf{vkList}^*$, $\mathsf{T}_\mathrm{b}[\mathsf{M}^*] = 0$, and $\widetilde{\mathsf{sig}}^*$ is a valid forgery on M^* under $(U_1, U_2) = \mathsf{H}_\mathsf{ck}(\mathsf{M}^*)$. Then, there exists $c^* = \mathsf{H}_\mathrm{c}(\widetilde{T}^*, \widetilde{\mathsf{pk}}^*, \mathsf{M}^*)$ in $\mathsf{T}_{\mathsf{H}_\mathrm{c}}$ s.t.

(a) $\widetilde{T}^* = \tilde{d}^*(U_1, U_2)^\top + \tilde{s}^*(G, H)^\top - c^* \cdot \widetilde{\mathsf{pk}}^*$,

(b) $\widetilde{\mathsf{pk}}^*$ is the aggregated key computed from vkList^*,

(c) $(U_1, U_2)^\top = \rho^*(G, H)^\top$.

Below, we show that $\mathcal{A}$ can make such a query with probability at most $(2Q_H + Q_S + 2)/p$.

To evaluate the probability, we rewrite the right-hand of the equation in (a) by $(G, H)^\top$ and $(X, Y)^\top$. Since (G, H, X, Y) in Game_6 is a non-DH tuple, $(G, H)^\top$ and $(X, Y)^\top$ are linearly independent. Then, we can denote the aggregated key $\widetilde{\mathsf{pk}}^*$ as $\phi^* (G, H)^\top + \psi^* (X, Y)^\top$ where $\phi^*, \psi^* \in \mathbb{Z}_p$. Substituting the above and (c) in the equation in (a), we have

$$\widetilde{T}^* = \left(\tilde{d}^* \rho^* + \tilde{s}^* - c^* \phi^*\right) (G, H)^\top - c^* \psi^* (X, Y)^\top . \tag{4.6}$$

Since $(G, H)^\top$ and $(X, Y)^\top$ are linearly independent, the values of coefficients $(\tilde{d}^* \rho^* + \tilde{s}^* - c^* \phi^*)$ and $c^* \psi^*$ which make Eq. (4.6) hold are uniquely determined when $(\widetilde{T}^*, \widetilde{\mathsf{pk}}^*, \mathsf{M}^*)$ is queried to H_c. Moreover, the values of ϕ^* and ψ^* are uniquely determined at the same point since the query includes $\widetilde{\mathsf{pk}}^*$. For the coefficient $c^* \psi^*$, c^* is determined by H_c and ψ^* is also determined by H_agg.

Here, we evaluate $\Pr[\mathsf{Game}_6 = 1]$. For $R \in \mathbb{G}^2$, let $\phi(R)$ and $\psi(R)$ be the elements in $\mathbb{Z}_p$ s.t. $R = \phi(R)(G, H)^\top + \psi(R)(X, Y)^\top$. Let E_agg be the event where there exists at least one random oracle query $\mathsf{H}_\mathsf{agg}((X', Y'), \mathsf{vkList}')$ s.t. $(X, Y) \in \mathsf{vkList}'$ and $\psi(\widetilde{\mathsf{pk}}') = 0$ for the aggregated key $\widetilde{\mathsf{pk}}'$ computed from vkList'. Then, we have

$$\begin{aligned}
&\Pr[\mathsf{Game}_6 = 1] \\
&= \Pr[\mathsf{Game}_6 = 1 \wedge \mathsf{E}_\mathrm{agg}] + \Pr[\mathsf{Game}_6 = 1 \wedge \overline{\mathsf{E}_\mathrm{agg}}] \\
&\leq \Pr[\mathsf{E}_\mathrm{agg}] + \Pr[\mathsf{Game}_6 = 1 \wedge \overline{\mathsf{E}_\mathrm{agg}}].
\end{aligned} \tag{4.7}$$

Also, let E_chal be the event where there exists at least one random oracle query $c' = \mathsf{H}_\mathrm{c}(\widetilde{T}', \widetilde{\mathsf{pk}}', \mathsf{M}')$ s.t. $\psi(\widetilde{\mathsf{pk}}') \neq 0$, $\mathsf{T}_\mathrm{b}[\mathsf{M}'] = 0$, and $\psi(\widetilde{T}') = c'\psi(\widetilde{\mathsf{pk}}')$. If $\mathsf{Game}_6 = 1$ occurs, $\mathsf{T}_\mathrm{b}[\mathsf{M}^*] = 0$ holds from the winning conditions. Also, if $\mathsf{Game}_6 = 1$ occurs, there exists at least one random oracle query to H_c making $\psi(\widetilde{T}^*) = c^*\psi(\widetilde{\mathsf{pk}}^*)$ hold since $\mathcal{A}$'s output satisfies Eq. (4.6). There is no query

$\mathsf{H}_{\mathsf{agg}}((X', Y'), \mathsf{vkList}')$ s.t. $(X, Y) \in \mathsf{vkList}'$ and $\psi(\widetilde{\mathsf{pk}}') = 0$ when $\overline{\mathsf{E}_{\text{agg}}}$ occurs. Thus, if $\overline{\mathsf{E}_{\text{agg}}}$ occurs, then $\psi(\widetilde{\mathsf{pk}}^*) \neq 0$ holds. Therefore, if $\mathsf{Game}_6 = 1$ and $\overline{\mathsf{E}_{\text{agg}}}$ occur, then E_{chal} occurs. Then, we have $\Pr[\mathsf{Game}_6 = 1 \wedge \overline{\mathsf{E}_{\text{agg}}}] \leq \Pr[\mathsf{E}_{\text{chal}}]$. Applying this fact to Eq. (4.7), we obtain

$$\Pr[\mathsf{Game}_6 = 1] \leq \Pr[\mathsf{E}_{\text{agg}}] + \Pr[\mathsf{E}_{\text{chal}}]. \tag{4.8}$$

First, we evaluate $\Pr[\mathsf{E}_{\text{agg}}]$. For an aggregate key $\widetilde{\mathsf{pk}}'$ computed from vkList', $\psi(\widetilde{\mathsf{pk}}') = \sum_{i=1}^{n'} t_i \psi(\mathsf{vkList}'[i])$ holds where n' is the number of the public keys in vkList' and $t_i = \mathsf{H}_{\mathsf{agg}}(\mathsf{vkList}'[i], \mathsf{vkList}')$. Since the challenger defines the value t_i for all $i \in [n']$ when vkList' is first queried to $\mathsf{H}_{\mathsf{agg}}$, $(t_i)_{i=1}^{n'}$ is uniformly chosen from $\mathbb{Z}_p^{n'}$ after $(\psi(\mathsf{vkList}'[i]))_{i=1}^{n'}$ is fixed. If vkList' includes (X, Y), there exists at least one i s.t. $\psi(\mathsf{vkList}'[i]) \neq 0$. Thus, per one query to $\mathsf{H}_{\mathsf{agg}}$, $\sum_{i=1}^{n'} t_i \psi(\mathsf{vkList}'[i]) = 0$ holds with probability at most $1/p$. Since at most $Q_H + Q_S + 1$ public key lists appear in $\mathsf{T}_{\mathsf{H}_{\mathsf{agg}}}$, we obtain

$$\Pr[\mathsf{E}_{\text{agg}}] \leq \frac{Q_H + Q_S + 1}{p}. \tag{4.9}$$

Next, we evaluate $\Pr[\mathsf{E}_{\text{chal}}]$. Let $\mathsf{E}_{\text{chal},j}$ be the event where the j-th random oracle query $c'_j = \mathsf{H}_{\text{c}}(\widetilde{T}'_j, \widetilde{\mathsf{pk}}'_j, \mathsf{M}'_j)$ satisfies following conditions.

$$\mathsf{E}_{j,1} : \psi(\widetilde{\mathsf{pk}}'_j) \neq 0,\ \mathsf{E}_{j,2} : \mathsf{T}_{\text{b}}[\mathsf{M}'_j] = 0,\ \text{and}\ \mathsf{E}_{j,3} : \psi(\widetilde{T}'_j) = c'_j \psi(\widetilde{\mathsf{pk}}'_j).$$

Note that there are at most $Q_H + 1$ queries $\mathsf{H}_{\text{c}}(\widetilde{T}', \widetilde{\mathsf{pk}}', \mathsf{M}')$ s.t. $\mathsf{T}_{\text{b}}[\mathsf{M}'] = 0$. From this fact and the union bound, we have

$$\begin{aligned}
\Pr[\mathsf{E}_{\text{chal}}] &\leq \sum_{i=1}^{Q_H+1} \Pr[\mathsf{E}_{\text{chal},j}] \\
&= \sum_{i=1}^{Q_H+1} \Pr[\mathsf{E}_{j,1} \wedge \mathsf{E}_{j,2} \wedge \mathsf{E}_{j,3}] \\
&\leq \sum_{i=1}^{Q_H+1} \Pr[\mathsf{E}_{j,3} | \mathsf{E}_{j,1} \wedge \mathsf{E}_{j,2}].
\end{aligned} \tag{4.10}$$

As described previously, when $(\widetilde{T}'_j, \widetilde{\mathsf{pk}}'_j, \mathsf{M}'_j)$ is queried to H_{c}, the value of $\psi(\widetilde{T}'_j)$ and $\psi(\widetilde{\mathsf{pk}}'_j)$ are fixed. Also, conditioned on $\mathsf{E}_{j,1}$, $\psi(\widetilde{\mathsf{pk}}'_j) \neq 0$ holds. Thus, conditioned on $\mathsf{E}_{j,1}$ and $\mathsf{E}_{j,2}$, before c'_j is chosen, c'_j making $\psi(\widetilde{T}'_j) = c'_j \psi(\widetilde{\mathsf{pk}}'_j)$ hold is determined uniquely. Since c'_j is uniformly chosen from $\mathbb{Z}_p$

independently of the j-th random oracle query, $\Pr[\mathsf{E}_{j,3}|\mathsf{E}_{j,1} \wedge \mathsf{E}_{j,2}]$ is at most $1/p$. From this and Eq. (4.10), we have

$$\Pr[\mathsf{E}_{\text{chal}}] \leq \sum_{i=1}^{Q_H+1} \frac{1}{p} = \frac{Q_H+1}{p}. \tag{4.11}$$

From Eqs. (4.8), (4.9) and (4.11), we obtain

$$\Pr[\mathsf{Game}_6 = 1] \leq \frac{2Q_H + Q_S + 2}{p}.$$

This completes the proof. □

4.6 Improved Scheme HBMSDDH-2

In this section, we improve the HBMSDDH-1 in Section 4.2 to be 2-MS-UF-1 . To achieve this goal, we subtly modify the original scheme HBMSDDH-1. Specifically, we add the aggregated key $\widetilde{\mathsf{pk}}$ to the input of the hash function H_{ck}. Then, we can show that the modified scheme HBMSDDH-2 satisfies the slightly strong unforgeability described in Section 2.4.4. Below, we provide the intuition of this improvement.

We need to modify the original scheme to be 2-MS-UF-1 because it is insecure under the winning condition $(\mathsf{vkList}^*, \mathsf{M}^*) \notin \mathsf{Q}_\mathsf{M}$. Indeed, we can construct an adversary that generates a forgery satisfying such the winning condition. Under such the winning condition, an adversary is allowed to forge the multi-signature on M^* under the commitment key $(U_1, U_2) = \mathsf{H}_{\mathsf{ck}}(\mathsf{M}^*)$ already used in the signing queries. This means that an adversary is allowed to reuse the commitment key. For our scheme, the ROS attack is prevented by not reusing commitment keys. Namely, the winning condition makes the ROS attack feasible. Therefore, we need to modify HBMSDDH-1 to be slightly strongly unforgeable.

The naive approach is adding vkList to the input of H_{ck}, but such a scheme no longer supports key aggregation. This modification makes the reuse of the commitment key infeasible. However, the verification algorithm no longer can verify a multi-signature without vkList. Thus, the verification algorithm always needs to take vkList as input. This makes the key aggregation meaningless.

To maintain the advantage of the key aggregation, we add $\widetilde{\mathsf{pk}}$ to the input of H_{ck}, instead of vkList. This modification can prevent the reuse of the commitment key without compromising the key aggregation. To reuse the commitment key, an adversary needs to find distinct two public key lists that lead to the same aggregated key. However, the aggregated key is deterministically computed from vkList and $\mathsf{H_{agg}}$. Also, the output of $\mathsf{H_{agg}}$ for each vkList is uniformly generated by the random oracle after vkList is fixed via the signing query. Then, we can prove that the probability of an adversary finding such two public key lists is negligible.

4.6.1 Construction of HBMSDDH-2

We give the construction of HBMSDDH-2 in Fig. 4.10. This is almost the same as HBMSDDH-1. The difference is that the hash function $\mathsf{H_{ck}}$ takes as input $(\mathsf{M}, \widetilde{\mathsf{pk}})$. We should note that this modification does not affect the signature size, communication complexity, and computational complexity. We omit the proof of the correctness of HBMSDDH-2 since it is easy to check it by following the proof of Theorem 12.

4.6.2 Security Proof of HBMSDDH-2

Here, we show that HBMSDDH-2 satisfies the slightly strong unforgeability defined in Section 2.4.4.

Theorem 14. *If $\mathbb{G}$ is a (t, ϵ)-DDH group, then* HBMSDDH-2 *is $(t_{\mathcal{A}}, Q_S, Q_H, N, \epsilon_{\mathcal{A}})$-2-MS-UF-1 s.t.*

$$\epsilon_{\mathcal{A}} \geq e(Q_S + 1)\left(2\epsilon + \frac{2Q_H + Q_S + 2}{p}\right) + \frac{(Q_H + Q_S + 1)^2}{p} \text{ and}$$

$$t_{\mathcal{A}} \leq \min(t_1, t_2) \text{ where}$$

$$t_1 = t - (2NQ_H + 6Q_S N + 4Q_S + 2N + 12)t_{\mathrm{mul}} - O(Q_H N + Q_S N),$$

$$t_2 = t - (2NQ_H + 6Q_S N + 2Q_S + 2N + 8)t_{\mathrm{mul}} - O(Q_H N + Q_S N),$$

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

Proof of Theorem 14. First, we prepare two notations. Let us denote the time for a scalar multiplication in $\mathbb{G}$ by t_{mul}. We write $\Pr[\mathsf{Game}_i = 1]$ to mean the probability that a forger wins the game Game_i.

Let $\mathcal{A}$ be an adversary that $(t_{\mathcal{A}}, Q_S, Q_H, N, \epsilon_{\mathcal{A}})$-breaks the 2-MS-UF-1 of HBMSDDH-2. For $\mathcal{A}$, we consider a sequence of games where the first hybrid Game_0 is the slightly strong unforgeability game in Fig. 2.10 for HBMSDDH-2. Game_0 is shown in Fig. 4.11. Because Game_0 is the unforgeability game of HBMSDDH-2, we have $\Pr[\mathsf{Game}_0 = 1] \geq \epsilon_{\mathcal{A}}$.

Now we change Game_0. We consider the similar sequence of games to that in the proof of Theorem 14.

$\underline{\mathsf{Game}_1}$**:** In this game, the challenger makes the random oracle query $\mathsf{H}_{\mathsf{ck}}(\mathsf{M}, \widetilde{\mathsf{pk}})$ at the beginning of both the random oracle H_{c} and the signing oracle $\mathcal{O}_{\mathsf{Sign}_1^{(2)}}$. This modification is as same as that we made in Game_1 in the proof of Theorem 14. Since an adversary cannot detect this modification, we have

$$\Pr[\mathsf{Game}_0 = 1] = \Pr[\mathsf{Game}_1 = 1].$$

$\underline{\mathsf{Game}_2}$**:** In this game, the challenger defines $\mathsf{H}_{\mathsf{agg}}((X_i, Y_i), \mathsf{vkList})$ for all $(X_i, Y_i) \in \mathsf{vkList}$. This modification is as same as that we made in Game_6 in the proof of Theorem 14. Since the challenger gives $\mathcal{A}$ only $\mathsf{T}_{\mathsf{H}_{\mathsf{agg}}}[(X, Y), \mathsf{vkList}]$, where $((X, Y), \mathsf{vkList})$ is queried, this change does not affect the probability of $\mathcal{A}$ winning the game. Thus, we have

$$\Pr[\mathsf{Game}_1 = 1] = \Pr[\mathsf{Game}_2 = 1].$$

$\underline{\mathsf{Game}_3}$**:** In this game, the challenger adds an abort condition in $\mathsf{H}_{\mathsf{agg}}$. Specifically, it additionally initializes a table $\mathsf{T}_{\mathsf{agg}}[\cdot]$ at the beginning of the game. In $\mathsf{H}_{\mathsf{agg}}$, after defined $\mathsf{H}_{\mathsf{agg}}((X_i, Y_i), \mathsf{vkList})$ for all $(X_i, Y_i) \in \mathsf{vkList}$, it computes an aggregated key $\widetilde{\mathsf{pk}}$ from the queried vkList if vkList includes the challenge key. Then, it checks $\mathsf{T}_{\mathsf{agg}}[\widetilde{\mathsf{pk}}] = \perp$. If it is true, it assigns $\mathsf{T}_{\mathsf{agg}}[\widetilde{\mathsf{pk}}] \leftarrow \mathsf{vkList}$ and continues the game. Otherwise, it aborts the game.

Now we evaluate the probability of the challenger aborting the game. It aborts the game when the distinct vkList_1 and vkList_2 lead to the same aggregated key $\widetilde{\mathsf{pk}}$. Let us denote this event by $\mathsf{E}_{\mathsf{bad}}$. Note that both lists include the challenge key. Let us consider the two types of vkList. The first type is that, for all $(X_i, Y_i) \in \mathsf{vkList}$, (G, H, X_i, Y_i) is a DH-tuple. The second type is that there exists at least one (X_i, Y_i) such that (G, H, X_i, Y_i) is a non-DH tuple. Since $(t_i)_i$ is uniformly chosen from $\mathbb{Z}_p^{|\mathsf{vkList}|}$ by $\mathsf{H}_{\mathsf{agg}}$ after vkList is fixed, the aggregated key computed from the first type is uniformly distributed in the span of $(G, H)^\top$, and that computed from the second type is uniformly distributed over $\mathbb{G}^2$. Thus, when both vkList_1 and vkList_2 are first type, the probability of collision of the aggregated keys computed from them is maximized and is at most $1/p$. Since at most $Q_H + Q_S + 1$ public key lists are queried to $\mathsf{H}_{\mathsf{agg}}$, we have $\Pr[\mathsf{E}_{\mathsf{bad}}] \leq (Q_H + Q_S + 1)^2/p$. Unless the challenger aborts the game, this game is identical to the previous game, we have

$$|\Pr[\mathsf{Game}_2 = 1] - \Pr[\mathsf{Game}_3 = 1]| \leq \frac{(Q_H + Q_S + 1)^2}{p}.$$

$\underline{\mathsf{Game}_4}$: In this game, the challenger partition the outputs (U_1, U_2) of $\mathsf{H}_{\mathsf{ck}}(\mathsf{M}, \widetilde{\mathsf{pk}})$ into two groups by following a biased coin $b_K \in \{0, 1\}$ and aborts the game if a bit b_K corresponding to (U_1, U_2) used in a signing query is 0 at the beginning of the second round. Moreover, it additionally checks the condition $\mathsf{T}_\mathrm{b}[\mathsf{M}^*, \widetilde{\mathsf{pk}}^*] = 0$, where $\widetilde{\mathsf{pk}}^*$ is the aggregated key computed from vkList. Specifically, it additionally initializes a table $\mathsf{T}_\mathrm{b}[\cdot] \leftarrow \perp$ at the beginning of the game. In H_{ck}, it firstly chooses a bit $b_K \in \{0, 1\}$ which becomes 1 with probability $\delta = Q_S/(Q_S + 1)$ and assigns $\mathsf{T}_\mathrm{b}[\mathsf{M}, \widetilde{\mathsf{pk}}] \leftarrow b_K$. Note that the way to generate (U_1, U_2) is unchanged. In the signing oracle $\mathcal{O}_{\mathsf{Sign}_2^{(2)}}$, it aborts the game if $\mathsf{T}_\mathrm{b}[\mathsf{M}, \widetilde{\mathsf{pk}}] = 0$ holds. Otherwise, it continues the game. At the end of the game, it checks the condition $\mathsf{T}_\mathrm{b}[\mathsf{M}^*, \widetilde{\mathsf{pk}}^*] = 0$ in addition to other conditions.

Now we relate the advantage of an adversary for this game to that for the previous game. Let us consider the following three events for this game.

E_1: The event where the game does not terminate in the signing oracle $\mathcal{O}_{\mathsf{Sign}_2^{(2)}}$.

E_2: The event where the output of $\mathcal{A}$ satisfies the added condition $\mathsf{T}_\mathrm{b}[\mathsf{M}^*, \widetilde{\mathsf{pk}}^*] = 0$.

E_3: The event where the output of $\mathcal{A}$ satisfies the winning conditions as same as Game_2.

Then, we have

$$\begin{aligned}\Pr[\mathsf{Game}_3 = 1] &= \Pr[\mathsf{E}_1 \wedge \mathsf{E}_2 \wedge \mathsf{E}_3] \\ &= \Pr[\mathsf{E}_1] \Pr[\mathsf{E}_3 | \mathsf{E}_1] \Pr[\mathsf{E}_2 | \mathsf{E}_1 \wedge \mathsf{E}_3].\end{aligned}$$

First, we evaluate $\Pr[\mathsf{E}_1]$. The game aborts in $\mathcal{O}_{\mathsf{Sign}_2^{(2)}}$ if $\mathsf{T}_\mathrm{b}[\mathsf{M}, \widetilde{\mathsf{pk}}] = 0$ holds for a queried message and an aggregated key computed from the queried public key list. Thus, E_1 occurs when $\mathsf{T}_\mathrm{b}[\mathsf{M}, \widetilde{\mathsf{pk}}] = 1$ holds for all messages and public key lists queried to the signing oracle. Since the responses of the random oracles and the responses of the signing oracle $\mathcal{O}_{\mathsf{Sign}_1^{(2)}}$ leak no information on the value of $\mathsf{T}_\mathrm{b}[\mathsf{M}, \widetilde{\mathsf{pk}}]$ for any m and $\widetilde{\mathsf{pk}}$, $\mathcal{A}$ can know $\mathsf{T}_\mathrm{b}[\mathsf{M}, \widetilde{\mathsf{pk}}]$ only when it observes whether the game continues or not. Also, $\mathcal{A}$ can only know $\mathsf{T}_\mathrm{b}[\mathsf{M}, \widetilde{\mathsf{pk}}] = 1$ for all messages and public key lists queried to $\mathcal{O}_{\mathsf{Sign}_2^{(2)}}$ as long as the game continues.

Therefore, the probability that $\mathsf{T}_{\rm b}[\mathsf{M}, \widetilde{\mathsf{pk}}] = 0$ holds for each queried message and public key list is equal to $(1-\delta)$. In consequence, because $\mathcal{A}$ can make at most Q_S signing queries, we have $\Pr[\mathsf{E}_1] \geq \delta^{Q_S}$. Setting $\delta = Q_S/(Q_S+1)$, we have $\Pr[\mathsf{E}_1] \geq \delta^{Q_S} \geq 1/e$. The last inequality holds because of the fact that $(1+1/Q_S)^{Q_S} < e$ for $Q_S > 0$.

Next, we evaluate $\Pr[\mathsf{E}_3|\mathsf{E}_1]$. Conditioned on E_1, this game does not terminate, and the distribution of the view of $\mathcal{A}$ in this game is identical to the distribution of the view of $\mathcal{A}$ in the previous game. Thus, $\mathcal{A}$'s output in this game satisfies the winning conditions of the previous game with the same probability as in the previous game. Namely, we have $\Pr[\mathsf{E}_3|\mathsf{E}_1] = \Pr[\mathsf{Game}_3 = 1]$.

Finally, we evaluate $\Pr[\mathsf{E}_2|\mathsf{E}_1 \wedge \mathsf{E}_3]$. Conditioned on E_1 and E_3, since $(\mathsf{M}^*, \mathsf{vkList}^*)$ has never queried to the signing oracles. Moreover, the modification that we made in the previous game guarantees that the aggregated key $\widetilde{\mathsf{pk}}^*$ computed from vkList^* does not have a collision with other aggregated keys that appeared in $\mathsf{H}_{\mathsf{agg}}$. Thus, $\mathcal{A}$ cannot know $\mathsf{T}_{\rm b}[\mathsf{M}^*, \widetilde{\mathsf{pk}}^*]$. Then, $\Pr[\mathsf{E}_2|\mathsf{E}_1 \wedge \mathsf{E}_3] = (1-\delta)$ holds. Setting $\delta = Q_S/(Q_S+1)$, we obtain $\Pr[\mathsf{E}_2|\mathsf{E}_1 \wedge \mathsf{E}_3] = 1/(Q_S+1)$. Combining all bounds, we obtain

$$\Pr[\mathsf{Game}_3 = 1] \leq e(Q_S+1)\Pr[\mathsf{Game}_4 = 1].$$

<u>Game_5:</u> In this game, the challenger modifies how it generates (U_1, U_2) in $\mathsf{H}_{\mathsf{ck}}(\mathsf{M}, \widetilde{\mathsf{pk}})$ corresponding to a value $\mathsf{T}_{\rm b}[\mathsf{M}, \widetilde{\mathsf{pk}}]$. This modification is the same as the modification that we made in Game_3 in the proof of Theorem 14. Specifically, it additionally initializes a table $\mathsf{T}_{\rm td}[\cdot] \leftarrow \bot$ at the beginning of the game. In H_{ck}, instead of $(U_1, U_2) \xleftarrow{\$} \mathbb{G}^2$, it chooses $\rho \xleftarrow{\$} \mathbb{Z}_p$, computes $(U_1, U_2)^\top \xleftarrow{\$} \rho(G, H)^\top$ when $\mathsf{T}_{\rm b}[\mathsf{M}, \widetilde{\mathsf{pk}}] = 0$, and computes $(U_1, U_2)^\top \xleftarrow{\$} \rho(G, H)^\top + (X, Y)^\top$ when $\mathsf{T}_{\rm b}[\mathsf{M}, \widetilde{\mathsf{pk}}] = 1$. Then, it assigns $\mathsf{T}_{\rm td}[\mathsf{M}] \leftarrow \rho$.

We construct an adversary $\mathcal{B}$ against the DDH problem that internally runs $\mathcal{A}$ as same as the proof of Lemma 3. The description of $\mathcal{B}$ is the same as $\mathcal{B}$ in the proof of Lemma 3 except for the parts related to the input of H_{ck}.

We evaluate the running time $t_{\mathcal{B}}$ of $\mathcal{B}$. The running time of $\mathcal{B}$ is not equal to that of $\mathcal{B}$ in the proof of Lemma 3 because of the modification that we made in Game_3. For the random oracle queries, $\mathsf{H}_{\mathsf{agg}}$ takes longer time than H_{c} and H_{ck}. Responding to one query to $\mathsf{H}_{\mathsf{agg}}$ requires $2N$ scalar multiplication and $O(N)$ other non-cryptographic operations. Combining this and the argument in the proof of Lemma 3, we have $t_{\mathcal{B}} \leq t_{\mathcal{A}} + (2NQ_H + 6Q_SN + 4Q_S + 2N + 12)t_{\text{mul}} + O(Q_HN + Q_SN)$.

Now we evaluate the advantage of $\mathcal{B}$. Since the difference in the input of H_{ck} does not affect the advantage of $\mathcal{B}$, we have $\mathsf{Adv}^{\text{ddh}}_{\mathcal{B}}(1^\lambda) = |\Pr[\mathsf{Game}_4 = 1] - \Pr[\mathsf{Game}_5 = 1]|$.

By assuming that $\mathbb{G}$ is a (t, ϵ)-DDH group, for $\mathcal{B}$ such that $t_{\mathcal{B}} \leq t$, $\mathsf{Adv}^{\text{ddh}}_{\mathcal{B}}(1^\lambda) \leq \epsilon$ holds. Since $t_{\mathcal{B}} \leq t_{\mathcal{A}} + (2NQ_H + 6Q_SN + 4Q_S + 2N + 12)t_{\text{mul}} + O(Q_HN + Q_SN)$ and $\mathsf{Adv}^{\text{ddh}}_{\mathcal{B}}(1^\lambda) = |\Pr[\mathsf{Game}_4 = 1] - \Pr[\mathsf{Game}_5 = 1]|$, if $\mathbb{G}$ is a (t, ϵ)-DDH group, for $\mathcal{A}$ such that $t_{\mathcal{A}} \leq t - (2NQ_H + 6Q_SN + 4Q_S + 2N + 12)t_{\text{mul}} - O(Q_HN + Q_SN)$, the following inequality holds.

$$|\Pr[\mathsf{Game}_4 = 1] - \Pr[\mathsf{Game}_5 = 1]| \leq \epsilon.$$

$\underline{\mathsf{Game}_6}$**:** In this game, the challenger modifies how it generates protocol messages and partial signatures. This modification is the same as the modification that we made in Game_4 in the proof of Theorem 14.

To evaluate the advantages of an adversary in this game, we construct an adversary $\mathcal{B}$ against the game $\mathsf{Game}^{\text{eqv}}$ in Lemma 4. The description of $\mathcal{B}$ is the same as $\mathcal{B}$ in the proof of Lemma 6 except for the parts related to the inputs of H_{ck}. Since the difference in the input of H_{ck} does not affect the advantage of $\mathcal{B}$ compared to $\mathcal{B}$ in the proof of Lemma 6, due to Lemma 4, we have

$$\Pr[\mathsf{Game}_5 = 1] = \Pr[\mathsf{Game}_6 = 1].$$

$\underline{\mathsf{Game}_7}$**:** In this game, the challenger changes how it generates the challenge key. This modification is the same as the modification that we made in Game_5 in the proof of Theorem 14.

We construct an adversary $\mathcal{B}$ against the DDH problem that internally runs $\mathcal{A}$ as same as the proof of Lemma 7. The description of $\mathcal{B}$ is the same as $\mathcal{B}$ in the proof of Lemma 7 except for the parts related to the input of H_{ck}. The running time of $\mathcal{B}$ is not equal to that of $\mathcal{B}$ in the proof of Lemma 7 because of the modification that we made in Game_3. For the random oracle queries, $\mathsf{H}_{\mathsf{agg}}$ takes a longer time than H_{c} and H_{ck}. Responding to one query to $\mathsf{H}_{\mathsf{agg}}$ requires $2N$ scalar multiplication and $O(N)$ other non-cryptographic operations. Combining this and the argument in the proof of Lemma 3, we have $t_{\mathcal{B}} \leq t_{\mathcal{A}} + (2NQ_H + 6Q_SN + 2Q_S + 2N + 8)t_{\text{mul}} + O(Q_HN + Q_SN)$. Moreover, since the difference in the input of H_{ck} does not affect the advantage of $\mathcal{B}$, we have $\mathsf{Adv}_{\mathcal{B}}^{\mathrm{ddh}}(1^\lambda) = |\Pr[\mathsf{Game}_6 = 1] - \Pr[\mathsf{Game}_7 = 1]|$.

By assuming that $\mathbb{G}$ is a (t, ϵ)-DDH group, for $\mathcal{B}$ such that $t_{\mathcal{B}} \leq t$, $\mathsf{Adv}_{\mathcal{B}}^{\mathrm{ddh}}(1^\lambda) \leq \epsilon$. Since $t_{\mathcal{B}} \leq t_{\mathcal{A}} + (2NQ_H + 6Q_SN + 2Q_S + 2N + 8)t_{\mathrm{mul}} + O(Q_HN + Q_SN)$ and $\mathsf{Adv}_{\mathcal{B}}^{\mathrm{ddh}}(1^\lambda) = |\Pr[\mathsf{Game}_6 = 1] - \Pr[\mathsf{Game}_7 = 1]|$, if $\mathbb{G}$ is a (t, ϵ)-DDH group, for $\mathcal{A}$ such that $t_{\mathcal{A}} \leq t - (2NQ_H + 6Q_SN + 2Q_S + 2N + 8)t_{\mathrm{mul}} - O(Q_HN + Q_SN)$, the following inequality holds.

$$|\Pr[\mathsf{Game}_6 = 1] - \Pr[\mathsf{Game}_7 = 1]| \leq \epsilon.$$

For the game Game_7, Lemma 8 holds even if we add $\widetilde{\mathsf{pk}}$ to the inputs of H_{ck}. Thus, we have

$$\Pr[\mathsf{Game}_7 = 1] \leq \frac{2Q_H + Q_S + 2}{p}.$$

By combining all arguments, if $\mathbb{G}$ is a (t, ϵ)-DDH group, for $\mathcal{A}$ such that

$$\begin{aligned} &t_{\mathcal{A}} \leq t - (2NQ_H + 6Q_SN + 4Q_S + 2N + 12)t_{\mathrm{mul}} - O(Q_HN + Q_SN), \\ \text{and } &t_{\mathcal{A}} \leq t - (2NQ_H + 6Q_SN + 2Q_S + 2N + 8)t_{\mathrm{mul}} - O(Q_HN + Q_SN), \end{aligned}$$

the following inequalities holds.

$$\begin{aligned} \epsilon_{\mathcal{A}} &= \Pr[\mathsf{Game}_2 = 1] \\ &\leq \Pr[\mathsf{Game}_3 = 1] + \frac{(Q_H + Q_S + 1)^2}{p} \\ &\leq e(Q_S + 1)\Pr[\mathsf{Game}_4 = 1] + \frac{(Q_H + Q_S + 1)^2}{p} \\ &\leq e(Q_S + 1)(\epsilon + \Pr[\mathsf{Game}_5 = 1]) + \frac{(Q_H + Q_S + 1)^2}{p} \\ &\leq e(Q_S + 1)(2\epsilon + \Pr[\mathsf{Game}_7 = 1]) + \frac{(Q_H + Q_S + 1)^2}{p} \\ &\leq e(Q_S + 1)\left(2\epsilon + \frac{2Q_H + Q_S + 2}{p}\right) + \frac{(Q_H + Q_S + 1)^2}{p}. \end{aligned}$$

Therefore, if $\mathbb{G}$ is a (t, ϵ)-DDH group, then $\mathsf{HBMSDDH\text{-}2}$ is $(t_{\mathcal{A}}, Q_S, Q_H, N, \epsilon_{\mathcal{A}})$-2-MS-UF-1 s.t.

$$\epsilon_{\mathcal{A}} \geq e(Q_S + 1)\left(2\epsilon + \frac{2Q_H + Q_S + 2}{p}\right) + \frac{(Q_H + Q_S + 1)^2}{p} \text{ and}$$

$t_{\mathcal{A}} \leq \min(t_1, t_2)$ where

$$
\begin{aligned}
t_1 &= t - (2NQ_H + 6Q_SN + 4Q_S + 2N + 12)t_{\text{mul}} - O(Q_HN + Q_SN), \\
t_2 &= t - (2NQ_H + 6Q_SN + 2Q_S + 2N + 8)t_{\text{mul}} - O(Q_HN + Q_SN).
\end{aligned}
$$

This completes the proof. □

$\mathsf{Setup}(1^\lambda)$:

$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$

$H \xleftarrow{\$} \mathbb{G}$

Select $\mathsf{H_c}$ // $\mathsf{H_c} : \{0,1\}^* \to \mathbb{Z}_p$

Select $\mathsf{H_{ck}}$ // $\mathsf{H_{ck}} : \{0,1\}^* \to \mathbb{G}^2$

Select $\mathsf{H_{agg}}$ // $\mathsf{H_{agg}} : \{0,1\}^* \to \mathbb{Z}_p$

return $\mathsf{par} = (\mathbb{G}, p, G, H, \mathsf{H_c}, \mathsf{H_{ck}}, \mathsf{H_{agg}})$.

$\mathsf{KeyGen}(\mathsf{par}) \to (\mathsf{pk}, \mathsf{sk})$:

$x \xleftarrow{\$} \mathbb{Z}_p$

$(X, Y)^\top \leftarrow x(G, H)^\top$

$\mathsf{pk} \leftarrow (X, Y)$

$\mathsf{sk} \leftarrow x$

return $(\mathsf{pk}, \mathsf{sk})$

$\mathsf{Agg}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, (\mathsf{pm}_i, \mathsf{psig}_i)_{i \in [N]})$:

parse $(X_i, Y_i)_{i \in [N]} \leftarrow \mathsf{vkList}$

parse $(T_i, d_i, s_i)_{i \in [N]} \leftarrow (\mathsf{pm}_i, \mathsf{psig}_i)_{i \in [N]}$

for $i \in [N]$ **do**

  $t_i \leftarrow \mathsf{H_{agg}}((X_i, Y_i), \mathsf{vkList})$

$\widetilde{\mathsf{pk}} \leftarrow \sum_{i=1}^{N} t_i (X_i, Y_i)^\top$

$\widetilde{T} \leftarrow \sum_{i=1}^{N} T_i$

$c \leftarrow \mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})$

$\tilde{d} \leftarrow \sum_{i=1}^{N} d_i \mod p$

$\tilde{s} \leftarrow \sum_{i=1}^{N} s_i \mod p$

$\widetilde{\mathsf{sig}} \leftarrow (c, \tilde{d}, \tilde{s})$

return $\widetilde{\mathsf{sig}}$

$\mathsf{Sign}_1^{(2)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i)$:

parse $(X_j, Y_j)_{j \in [N]} \leftarrow \mathsf{vkList}$

for $j \in [N]$ **do**

  $t_j \leftarrow \mathsf{H_{agg}}((X_j, Y_j), \mathsf{vkList})$

$\widetilde{\mathsf{pk}} \leftarrow \sum_{j=1}^{N} t_j (X_j, Y_j)^\top$

$(U_1, U_2) \leftarrow \mathsf{H_{ck}}(\mathsf{M}, \widetilde{\mathsf{pk}})$

$r_i, d_i \xleftarrow{\$} \mathbb{Z}_p$

$T_i \leftarrow d_i (U_1, U_2)^\top + r_i (G, H)^\top$

$\mathsf{pm}_i \leftarrow T_i$

$\mathsf{st}_i \leftarrow (r_i, d_i, t_i, T_i, \widetilde{\mathsf{pk}})$

return $(\mathsf{pm}_i, \mathsf{st}_i)$

$\mathsf{Sign}_2^{(2)}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, i, \mathsf{sk}_i, \mathsf{st}_i, (\mathsf{pm}_j)_{j \in [N] \setminus \{i\}})$:

parse $(T_j)_{j \in [N] \setminus \{i\}} \leftarrow (\mathsf{pm}_j)_{j \in [N] \setminus \{i\}}$

parse $(r_i, d_i, t_i, T_i, \widetilde{\mathsf{pk}}) \leftarrow \mathsf{st}_i$

$\widetilde{T} \leftarrow \sum_{j=1}^{N} T_j$

$c \leftarrow \mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})$

$s_i \leftarrow x_i t_i c + r_i \mod p$

$\mathsf{psig}_i \leftarrow (d_i, s_i)$

return psig_i

$\mathsf{Verify}(\mathsf{par}, \mathsf{vkList}, \mathsf{M}, \widetilde{\mathsf{sig}})$:

parse $(X_i, Y_i)_{i \in [N]} \leftarrow \mathsf{vkList}$

parse $(c, \tilde{d}, \tilde{s}) \leftarrow \widetilde{\mathsf{sig}}$

for $i \in [N]$ **do**

  $t_i \leftarrow \mathsf{H_{agg}}((X_i, Y_i), \mathsf{vkList})$

$\widetilde{\mathsf{pk}} \leftarrow \sum_{i=1}^{N} t_i (X_i, Y_i)^\top$

$(U_1, U_2) \leftarrow \mathsf{H_{ck}}(\mathsf{M}, \widetilde{\mathsf{pk}})$

$\widetilde{T} \leftarrow \tilde{d}(U_1, U_2)^\top + \tilde{s}(G, H)^\top - c \cdot \widetilde{\mathsf{pk}}$

if $[\![c = \mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})]\!]$ **then**

  return 1

return 0

Figure 4.10: The construction of HBMSDDH-2. The differences from HBMSDDH-1 are highlighted in blue. $\mathsf{H_c}, \mathsf{H_{ck}}$, and $\mathsf{H_{agg}}$ are modeled as random oracles. N is tne number of the signers in vkList.

$\mathsf{Game}_0 = \mathsf{Game}^{\mathsf{ms^2\text{-}uf1}}_{\mathsf{MS}^{(2)},\mathcal{A}}(1^\lambda, N)$

$\mathsf{Q_M} \leftarrow \emptyset, \mathsf{Q_{st}}[\cdot] \leftarrow \bot$
$\mathsf{T_{H_c}}[\cdot] \leftarrow \bot, \mathsf{T_{H_{ck}}}[\cdot] \leftarrow \bot, \mathsf{T_{H_{agg}}}[\cdot] \leftarrow \bot$
$(\mathbb{G}, p, G) \xleftarrow{\$} \mathsf{GrGen}(1^\lambda)$
$H \xleftarrow{\$} \mathbb{G}$
$x \xleftarrow{\$} \mathbb{Z}_p$
$(X, Y)^\top \leftarrow x(G, H)^\top$
$\mathsf{pk} \leftarrow (X, Y)$
$\mathsf{sk} \leftarrow x$
$(\mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}_1^{(2)}}, \mathcal{O}_{\mathsf{Sign}_2^{(2)}}, \mathsf{H_c}, \mathsf{H_{ck}}, \mathsf{H_{agg}}}(\mathsf{par}, \mathsf{pk})$
$\textbf{req } [\![\mathsf{pk} \in \mathsf{vkList}^*]\!] \wedge [\![|\mathsf{vkList}^*| \leq N]\!]\, [\![(\mathsf{vkList}^*, \mathsf{M}^*) \notin \mathsf{Q_M}]\!]$
$\textbf{return } \mathsf{Verify}(\mathsf{par}, \mathsf{vkList}^*, \mathsf{M}^*, \widetilde{\mathsf{sig}}^*)$

$\mathcal{O}_{\mathsf{Sign}_1^{(2)}}(\mathsf{sid}, \mathsf{vkList}, \mathsf{M})$

$\textbf{req } [\![\mathsf{pk} \in \mathsf{vkList}]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 1] = \bot]\!] \wedge [\![|\mathsf{vkList}| \leq N]\!]$
$\mathsf{HS_{sid}} \leftarrow \emptyset$
$N \leftarrow |\mathsf{vkList}|$
$\textbf{parse } (\mathsf{pk}_i)_{i \in [N]} \leftarrow \mathsf{vkList}$
$\textbf{for } i \in [N] \textbf{ do}$
$\quad \textbf{if } \mathsf{pk}_i = \mathsf{pk} \textbf{ then}$
$\quad\quad \mathsf{HS_{sid}} \leftarrow \mathsf{HS_{sid}} \cup \{i\}$
$\textbf{parse } (X_j, Y_j)_{j \in [N]} \leftarrow \mathsf{vkList}$
$\textbf{for } j \in [N] \textbf{ do}$
$\quad t_j \leftarrow \mathsf{H_{agg}}((X_j, Y_j), \mathsf{vkList})$
$\widetilde{\mathsf{pk}} \leftarrow \sum_{j=1}^{N} t_j (X_j, Y_j)^\top$
$(U_1, U_2) \leftarrow \mathsf{H_{ck}}(\mathsf{M}, \widetilde{\mathsf{pk}})$
$\textbf{for } i \in \mathsf{HS_{sid}} \textbf{ do}$
$\quad r_i, d_i \xleftarrow{\$} \mathbb{Z}_p$
$\quad T_i \leftarrow d_i (U_1, U_2)^\top + r_i (G, H)^\top$
$\quad \mathsf{pm}_i \leftarrow T_i$
$\quad \mathsf{st}_i \leftarrow (r_i, d_i, t_i, T_i, \widetilde{\mathsf{pk}})$
$\mathsf{Q_{st}}[\mathsf{sid}, 1] \xleftarrow{\$} (\mathsf{vkList}, \mathsf{M}, \mathsf{HS_{sid}}, (\mathsf{st}_i)_{i \in \mathsf{HS_{sid}}})$
$\textbf{return } (\mathsf{pm}_i)_{i \in \mathsf{HS_{sid}}}$

$\mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})$:

$\textbf{if } [\![\mathsf{T_{H_c}}[\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M}] = \bot]\!]$
$\quad c \xleftarrow{\$} \mathbb{Z}_p$
$\quad \mathsf{T_{H_c}}[\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M}] \leftarrow c$
$\textbf{return } \mathsf{T_{H_c}}[\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M}]$

$\mathsf{H_{ck}}(\mathsf{M}, \widetilde{\mathsf{pk}})$:

$\textbf{if } [\![\mathsf{T_{H_{ck}}}[\mathsf{M}, \widetilde{\mathsf{pk}}] = \bot]\!]$
$\quad (U_1, U_2) \xleftarrow{\$} \mathbb{G}^2$
$\quad \mathsf{T_{H_{ck}}}[\mathsf{M}, \widetilde{\mathsf{pk}}] \leftarrow (U_1, U_2)$
$\textbf{return } \mathsf{T_{H_{ck}}}[\mathsf{M}, \widetilde{\mathsf{pk}}]$

$\mathsf{H_{agg}}((X, Y), \mathsf{vkList})$:

$\textbf{if } [\![\mathsf{T_{H_{agg}}}[(X, Y), \mathsf{vkList}] = \bot]\!]$
$\quad t \xleftarrow{\$} \mathbb{Z}_p$
$\quad \mathsf{T_{H_{agg}}}[(X, Y), \mathsf{vkList}] \leftarrow t$
$\textbf{return } \mathsf{T_{H_{agg}}}[(X, Y), \mathsf{vkList}]$

$\mathcal{O}_{\mathsf{Sign}_2^{(2)}}(\mathsf{sid}, (\mathsf{pm}_j)_{j \in [|\mathsf{vkList}|] \setminus \mathsf{HS}})$

$\textbf{req } [\![\mathsf{Q_{st}}[\mathsf{sid}, 1] \neq \bot]\!] \wedge [\![\mathsf{Q_{st}}[\mathsf{sid}, 2] = \bot]\!]$
$(\mathsf{vkList}, \mathsf{M}, \mathsf{HS_{sid}}, (\mathsf{st}_i)_{i \in \mathsf{HS_{sid}}}) \leftarrow \mathsf{Q_{st}}[\mathsf{sid}, 1]$
$N \leftarrow |\mathsf{vkList}|$
$\textbf{parse } (T_j)_{j \in [N] \setminus \mathsf{HS}} \leftarrow (\mathsf{pm}_j)_{j \in [N] \setminus \mathsf{HS}}$
$\textbf{for } i \in \mathsf{HS_{sid}} \textbf{ do}$
$\quad \textbf{parse } (r_i, d_i, t_i, T_i, \widetilde{\mathsf{pk}}) \leftarrow \mathsf{st}_i$
$\widetilde{T} \leftarrow \sum_{j=1}^{N} T_j$
$c \leftarrow \mathsf{H_c}(\widetilde{T}, \widetilde{\mathsf{pk}}, \mathsf{M})$
$\textbf{parse } x \leftarrow \mathsf{sk}$
$\textbf{for } i \in \mathsf{HS_{sid}} \textbf{ do}$
$\quad s_i \leftarrow x t_i c + r_i \mod p$
$\quad \mathsf{psig}_i \leftarrow (d_i, s_i)$
$\mathsf{Q_{st}}[\mathsf{sid}, 2] \leftarrow (\mathsf{psig}_i)_{i \in \mathsf{HS_{sid}}}$
$\mathsf{Q_M} \leftarrow \mathsf{Q_M} \cup \{(\mathsf{vkList}, \mathsf{M})\}$
$\textbf{return } (\mathsf{psig}_i)_{i \in \mathsf{HS_{sid}}}$

Figure 4.11: The initial game Game_0, that identical to the slightly strong unforgeability game in Fig. 2.10 for HBMSDDH-2.

Chapter 5

Analysis of Efficiency

In this chapter, we compare our schemes with the related schemes and analyze the efficiency of them. Specifically, we estimate the required size of the underlying group for 128-bit security for all schemes and compare the signature size and communication complexity in concrete security. Moreover, we show the result of the implementation experiment of our scheme and analyze the computation time. We also evaluate the communication time under a certain communication model.

Road Maps. In Section 5.1, we compare our scheme with the related schemes in concrete security. In Section 5.2, we show and evaluate the result of our implementation experiment. In Section 5.3, we estimate the communication time considering a certain communication environment and evaluate it.

5.1 Comparison in Concrete Security

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, the DDH, the AOMDL, or the OMDL assumptions, e.g., MuSig2 [NRS21], DWMS [AB21], HBMS [BD21], LK [LK22], MuSig-DN [NRSW20], TZ [TZ23], mBCJ [DEF+19], PW-1 [PW23], and PW-2 [PW23].

We remark on the followings on HBMS and mBCJ. For HBMS, in [BD21], Bellare and Dai showed the security proof of HBMS both under the AGM and without using it. 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*. We call this scheme mBCJ-PPK. This is because the original mBCJ is proven secure in *the key verification model*.

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 5.1.1. Table 5.1 summarizes the comparison.

5.1.1 Estimation of the Underlying Group Size

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

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

Step 1. We obtained inequalities $\epsilon_P \geq B_\epsilon(\epsilon_\mathcal{A}, Q_S, Q_H, N, p)$ and $t_P \leq B_t(t_\mathcal{A}, Q_S, Q_H, N, p)$ from the security proof, where B_ϵ 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 $\epsilon_\mathcal{A}$ and $t_\mathcal{A}$ are the success probability and the running time of a forger, respectively.

Step 2. We derived the inequality $t_P/\epsilon_P \leq B_t(t_\mathcal{A}, Q_S, Q_H, N, p)/B_\epsilon(\epsilon_\mathcal{A}, Q_S, Q_H, N, p) =: B_{t/p}(t_\mathcal{A}, \epsilon_\mathcal{A}, Q_S, Q_H, N, p)$ from the previous step.

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

In Step 3, we assumed $t_{\mathrm{dl}}/\epsilon_{\mathrm{dl}} = t_{\mathrm{ddh}}/\epsilon_{\mathrm{ddh}} = t_{\mathrm{aomdl}}/\epsilon_{\mathrm{aomdl}} = t_{\mathrm{omdl}}/\epsilon_{\mathrm{omdl}} = \sqrt{p}$. This assumption is natural because of the following two facts. The first fact is that the best-known attack for solving the DDH problem, the AOMDL 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 [Pol78], which requires $O(\sqrt{p})$ 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 [LP20] to SHA-1 with complexity $2^{61.2}$ with a margin. We set $Q_S = 2^{30}$ for a large scenario as in [GHKP18]. We set $N = 2^{15}$ for a large-scale setting.[2]

Remarks for Estimation. We estimate $|p|_{128}$ according to the steps described above and show the results of this estimation in Column 8 in Table 5.1. Here, we should remark on the following points for this estimation.

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

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

For MuSig2-1, we suppose $\nu = 4$ where ν is a unique parameter. For MuSig2 and DWMS, we obtained B_ϵ and B_t from [BD21, Appendix A]. For HBMS-AGM, we obtained B_ϵ and B_t from [BD21, Theorem 7.1]. For LK, we obtained B_ϵ and B_t from [LK22, Theorem 4.1]. For B_t of this, we suppose $t_P = t_\mathcal{A}$ 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_ϵ and B_t from [BD21, Appendix A]. For B_ϵ 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_ϵ and B_t from [BD21, Theorem 3.2, 3.4, and 7.2]. For TZ, we obtained B_ϵ and B_t from [TZ23, Theorem 2]. For mBCJ-PPK, we obtain B_ϵ and B_t from Theorem 3. For PW-1 and PW-2, we obtained B_ϵ and B_t from [PW23, Theorem 3.5 and 3.3], respectively.

For MuSig2-2, DWMS, HBMS-AGM, LK, and PW-1 the results of their estimation of $|p|_{128}$ are 257, 258, or 260. 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 very small exceedance of the group size, whose effects on concrete security are small.

5.1.2 Comparison

We compare the efficiency of the related two-round multi-signature schemes in Table 5.1 under provably secure parameters. Here, since both of our schemes HBMSDDH-1 and HBMSDDH-2 achieve the same $|p|_{128}$, we compare HBMSDDH-1 to the related two-round schemes.

First, we compare our scheme HBMSDDH-1 to the schemes having large reduction losses which are proven secure without using the AGM, i.e., MuSig2-1, MuSig-DN, HBMS, TZ, and mBCJ-PPK. Among these schemes, HBMSDDH-1 has the most efficient signature size and communication complexity. More concretely, $|\widetilde{\mathsf{sig}}|_{128}$ of ours is reduced by about 22% from MuSig2-1 and MuSig-DN, about 60% from HBMS, and about 45% from TZ and mBCJ-PPK. 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 of our scheme without the rewinding technique. However, we should state that the DDH assumption is a stronger (not weaker) computational assumption than the DL assumption. For MuSig2-1, the AOMDL assumption is also stronger than the DL assumption. Multi-signatures of MuSig2-1 and MuSig-DN consist of only an element in $\mathbb{G}$ and an element in $\mathbb{Z}_p$, 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. For MuSig2-1 and TZ, the first round of signing protocols can be executed before a message to be signed is determined. The online communication complexity of these schemes is smaller than ours.

Next, we compare HBMSDDH-1 to the schemes proven secure in the AGM, i.e., MuSig2-2, DWMS,HBMS-AGM, and LK. The signature size and the communication complexity of these schemes are more efficient than ours. Concretely, $|\widetilde{\mathsf{sig}}|_{128}$ of our scheme is about 2.2 times longer than MuSig2-2 and DWMS and about 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.[3] 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, $|p|_{128}$ of ours is larger than the other schemes. Note that our scheme does not require the AGM. For MuSig2-2, and DWMS, the signature size is the most efficient among all the two-round schemes.

Finally, we compare our scheme to PW-1 and PW-2. To ensure 128-bit security, PW-1 can use P-256, and PW-2 and our scheme can use P-384. The signature size and communication complexity of our scheme are the most efficient among these schemes. The signature size of ours is reduced by about 67% and 40% from PW-1 and PW-2, respectively. The communication complexity of ours is reduced by about 57% and 41% from PW-1 and PW-2, respectively. PW-1 does not support key aggregation. All schemes are proven secure under the DDH assumption in the random oracle model. Thus, our scheme can be considered an improvement on PW-1 and PW-2 under provably secure parameters.

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 total communication complexity. On the other hand, considering the online-offline paradigm, the online communication complexity of our scheme is larger than that of MuSig2-1 and TZ. This shows a trade-off between the signature size and the online communication complexity. 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.

[3] HBMS-AGM can eliminate the reduction loss caused by the technique of Coron [Cor00] due to the AGM. For more details, see [BD21, Appendix I].

Table 5.1: Detailed performance comparison among two-round multi-signature schemes.

Scheme	Assumption	Model	Loss	Signature Size	Communication Complexity	Pub. Key Size	$\lvert p\rvert_{128}$ (bit) Curve	$\lvert\widetilde{\mathsf{sig}}\rvert_{128}$ (bit) $\lvert\widetilde{\mathsf{sig}}\rvert_{\mathrm{EC}}$ (bit)	$\lvert\mathrm{CC}\rvert_{128}$ (bit) $\lvert\mathrm{CC}\rvert_{\mathrm{EC}}$ (bit)	Key Agg.
MuSig2-2 [NRS21]	AOMDL	AGM, ROM	$O(1)$	$\lvert\mathbb{G}\rvert + \lvert\mathbb{Z}_p\rvert$	$2\lvert\mathbb{G}\rvert + \lvert\mathbb{Z}_p\rvert$	$\lvert\mathbb{G}\rvert$	257 P-256	515 513	773 770	Yes
DWMS [AB21]	OMDL	AGM, ROM	$O(1)$	$\lvert\mathbb{G}\rvert + \lvert\mathbb{Z}_p\rvert$	$2\lvert\mathbb{G}\rvert + \lvert\mathbb{Z}_p\rvert$	$\lvert\mathbb{G}\rvert$	257 P-256	515 513	773 770	Yes
HBMS-AGM [BD21]	DL	AGM, ROM	$O(1)$	$\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_p\rvert$	$\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_p\rvert$	$\lvert\mathbb{G}\rvert$	257 P-256	772 769	772 769	Yes
LK [LK22]	DL	AGM, NPROM	$O(1)$	$3\lvert\mathbb{Z}_p\rvert$	$\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_p\rvert$	$\lvert\mathbb{G}\rvert$	258 P-256	774 768	775 769	Yes
MuSig-DN [NRSW20]	DL, DDH, PRNG zk-SNARKs, PRF	ROM	$O(Q_H^3/\epsilon^3)$	$\lvert\mathbb{G}\rvert + \lvert\mathbb{Z}_p\rvert$	$2\lvert\mathbb{G}\rvert + \lvert\mathbb{Z}_p\rvert + \lvert\pi\rvert$	$\lvert\mathbb{G}\rvert$	740 Not Exist	1481 -	- -	Yes
MuSig2-1 [NRS21]	AOMDL	ROM	$O(Q_H^3/\epsilon^3)$	$\lvert\mathbb{G}\rvert + \lvert\mathbb{Z}_p\rvert$	$4\lvert\mathbb{G}\rvert + \lvert\mathbb{Z}_p\rvert$	$\lvert\mathbb{G}\rvert$	750 Not Exist	1501 -	3754 -	Yes
HBMS [BD21]	DL	ROM	$O(Q_S^4 Q_H^3/\epsilon^3)$	$\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_p\rvert$	$\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_p\rvert$	$\lvert\mathbb{G}\rvert$	986 Not Exist	2959 -	2959 -	Yes
TZ [TZ23]	DL	ROM	$O(Q_H^3/\epsilon^3)$	$\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_p\rvert$	$4\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_p\rvert$	$\lvert\mathbb{G}\rvert$	742 Not Exist	2227 -	4456 -	Yes
mBCJ-PPK [DEF+19]	DL	ROM	$O(Q_S^2 Q_H/\epsilon)$	$\lvert\mathbb{G}\rvert + 3\lvert\mathbb{Z}_p\rvert$	$2\lvert\mathbb{G}\rvert + 3\lvert\mathbb{Z}_p\rvert$	$\lvert\mathbb{G}\rvert$	574 Not Exist	2297 -	2872 -	No
PW-1 [PW23]	DDH	ROM	$O(1)$	$6\lvert\mathbb{G}\rvert + 8\lvert\mathbb{Z}_p\rvert + N$	$6\lvert\mathbb{G}\rvert + 8\lvert\mathbb{Z}_p\rvert + 1$	$4\lvert\mathbb{G}\rvert$	260 P-256	3646+N 3590+N	3647 3591	No
PW-2 [PW23]	DDH	ROM	$O(Q_S)$	$5\lvert\mathbb{Z}_p\rvert$	$3\lvert\mathbb{G}\rvert + 4\lvert\mathbb{Z}_p\rvert$	$2\lvert\mathbb{G}\rvert$	322 P-384	1610 1920	2257 2691	Yes
HBMSDDH-1 HBMSDDH-2	DDH	ROM	$O(Q_S)$	$3\lvert\mathbb{Z}_p\rvert$	$2\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_p\rvert$	$2\lvert\mathbb{G}\rvert$	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 the reduction loss. Columns 5, 6 and 7 show the size of a multi-signature, elements sent in the signing protocol per a signer, and a public key, respectively. Column 8 shows the required underlying group size $|p|_{128}$ and the NIST standardized EC that enables a parameter choice with 128-bit security, which is called the recommended EC hereafter. Column 9 shows the signature sizes $|\widetilde{\mathsf{sig}}|_{128}$ and $|\widetilde{\mathsf{sig}}|_{\mathrm{EC}}$ under the $|p|_{128}$-bit EC group and the recommended EC, respectively. Column 10 shows the communication complexities $|\mathrm{CC}|_{128}$ and $|\mathrm{CC}|_{\mathrm{EC}}$ under the $|p|_{128}$-bit EC group and the recommended EC. Column 11 shows whether each scheme allows key aggregation. $\mathbb{G}$ and $\mathbb{Z}_p$ indicate the underlying group $\mathbb{G}$ of a prime order p and the ring of integers modulo p, respectively. We assume that the sizes of $|\mathbb{G}|$ and $|\mathbb{Z}_p|$ over a p-bit EC are $p+1$ and p bits, respectively. ROM and NPROM indicate the random oracle model and the non-programmable random oracle model. Q_H and Q_S indicate the number of random oracle queries and signing oracle queries, respectively. ϵ indicates the advantage of an adversary against the scheme. N indicates the number of signers. $|\pi|$ is the size of the zk-SNARK proof. For MuSig-DN, we write "-" in Column 10 because the size of $|\pi|$ considering concrete security is explicitly unknown.

5.2 Computation Time

In this section, we describe 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 5.2. In particular, we focus on HBMSDDH-1 because HBMSDDH-2 does not need additional computation and communication compared to the original one.

5.2.1 Environment and Setting

Environment. Our implementation is written in C++. We implemented our scheme by using the mcl library [Mit22] 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 5.2, 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 consider the case where the signer participated in the signing protocol aggregates the partial signatures. For details on the aggregation algorithm, see Remark 3. We measured $\mathsf{Sign}_1^{(2)}$ of the signing protocol in two phases. Specifically, one phase is computing the aggregated key $\widetilde{\mathsf{pk}}$ from a public key list vkList, and the other phase is computing other computations. For the verification, we measured the time for the verification algorithm without $\widetilde{\mathsf{pk}}$ shown in Section 4.2 and for the one given an aggregated key $\widetilde{\mathsf{pk}}$ instead of a public key list vkList.

5.2.2 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 time 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 $\mathsf{Sign}_1^{(2)}$ of the signing protocol for the computation of an aggregated key $\widetilde{\mathsf{pk}}$. By precomputing $\widetilde{\mathsf{pk}}$, $\mathsf{Sign}_1^{(2)}$ took only about 1 ms because it needs 4 scalar multiplications irrelevantly to N. Since there is no scalar multiplication in $\mathsf{Sign}_2^{(2)}$ and the aggregation by the signer participated in the signing protocol, they were completed within 0.2 ms even when $N = 100$.

For Verify without $\widetilde{\mathsf{pk}}$, 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 only 6 scalar multiplications by using $\widetilde{\mathsf{pk}}$, Verify with $\widetilde{\mathsf{pk}}$ 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.

5.2.3 Comparison

From a comparison with the computation time of PW-2, our scheme is more efficient than PW-2 when N is small or an aggregated key is pre-computed. Note that we adopted PW-2 as the peer for comparison because of the two reasons. The first reason is that it can be implemented under the same EC, i.e., P-384, for 128-bit security. The second reason is that other related schemes have no standardized EC for 128-bit security.

In comparison here, we only compare for the computation time of $\mathsf{Sign}_1^{(2)}$ of the signing protocol and the verification because of the two facts. The first fact is that the key generation algorithms of both schemes are identical. The second fact is that $\mathsf{Sign}_2^{(2)}$ and the aggregation by the signer participated in the signing protocol of both schemes have no scalar multiplication.

We estimate the computation time of PW-2 by using the result of the above measurement. Specifically, we assume that one scalar multiplication in $\mathbb{G}$ takes 0.25 ms. We estimate the computation time by multiplying the number of scalar multiplications and 0.25 ms.

In the cases where N is small or an aggregated key is pre-computed, our scheme is more efficient than PW-2. $\mathsf{Sign}_1^{(2)}$ of PW-2 except for the computation of an aggregated key takes about 2.8 ms because there are 11 scalar multiplications. Verify with $\widetilde{\mathsf{pk}}$ of PW-2 takes about 3.3 ms because it requires 13 scalar multiplications, which is more than twice as many as ours.

Additionally, our scheme is as efficient as PW-2 when N is large. This is because the time of computation of the aggregated key is dominant over the computation times of $\mathsf{Sign}_1^{(2)}$ and the verification. For example, the computation time of an aggregated key is about 64 ms for both schemes when N is 100. Then, the computation times of $\mathsf{Sign}_1^{(2)}$ and the verification for PW-2 are about 67 ms. There are only slight differences between the times of ours and PW-2.

5.3 Communication Time

In this section, we estimate the communication time of our scheme and PW-2 and compare them.

As the result of the estimation under the situation where each signer is connected to a hub by WAN, the latency is dominant even when $N = 100$ for both schemes. In other words, there is a small difference in the communication times between both schemes. Specifically, the communication time for each round of our scheme is about 61 ms, and the communication time for each round of PW-2 is about 62 ms. For both schemes, 60 ms of these communication times is the latency.

Here, we show how we derived the communication times of both schemes. We suppose the WAN environment with a bandwidth of 100 Mbps and a latency of 30 ms. In our signing protocol, in the first round, each signer sends 2 elements in $\mathbb{G}$ to the hub and receives $2(N-1)$ elements in $\mathbb{G}$ from the hub, and in the second round, it sends 2 elements in $\mathbb{Z}_p$ to the hub and receives $2(N-1)$ elements in $\mathbb{Z}_p$ from the hub. Then, when N is 100, the communication time for the first round is $770/(100 \times 10^3) + 30 + 770 \times 99/(100 \times 10^3) + 30 \approx 61$ ms, and that for the second round is $768/(100 \times 10^3) + 30 + 768 \times 99/(100 \times 10^3) + 30 \approx 61$ ms. In PW-2, in the first round, each signer sends 3 elements in $\mathbb{G}$ to the hub and receives $3(N-1)$ elements in $\mathbb{G}$ from the hub, and in the second round, it sends 4 elements in $\mathbb{Z}_p$ to the hub and receives $4(N-1)$ elements in $\mathbb{Z}_p$ from the hub. Then, when N is 100, the communication time for the first round is $1155/(100 \times 10^3) + 30 + 1155 \times 99/(100 \times 10^3) + 30 \approx 62$ ms, and that for the second round is $1536/(100 \times 10^3) + 30 + 1536 \times 99/(100 \times 10^3) + 30 \approx 62$ ms.

Table 5.2: 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.						
KeyGen	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.						
$\mathsf{Sign}_1^{(2)}$ (Computing $\widetilde{\mathsf{pk}}$)	1.4	2.5	5.0	8.0	29	64
$\mathsf{Sign}_1^{(2)}$ (Others)	1.0	1.1	1.1	1.1	1.1	1.2
$\mathsf{Sign}_2^{(2)}$	1.7×10^{-2}	2.0×10^{-2}	2.7×10^{-2}	3.5×10^{-2}	9×10^{-2}	1.7×10^{-1}
Aggregation	1.8×10^{-4}	2.2×10^{-4}	3.0×10^{-4}	4.1×10^{-4}	1×10^{-3}	2×10^{-3}
Verification.						
Verify without $\widetilde{\mathsf{pk}}$	3.0	4.1	6.9	9.6	31	66
Verify with $\widetilde{\mathsf{pk}}$	1.5	1.6	1.6	1.6	1.7	1.7

For the aggregation algorithm, we consider the situation where the signer participated in the signing protocol aggregates the partial signatures.

Chapter 6

Discussion

In this chapter, we discuss our results from both practical and theoretical aspects. We first review the research question and result of this thesis. Next, we describe the benefits of our scheme in practical terms. After that, we explain our results in theoretical terms. Finally, we mention open problems.

Review of Our Research Question and Result. In our research, we aim to construct a two-round multi-signature scheme that achieves reliable security and high efficiency without relying on the algebraic group model (AGM). In other words, we tried to construct a scheme whose signature size is smaller than related schemes without using the AGM under the provable secure parameters, which is derived by considering the reduction loss. As the result of this research, we proposed two two-round multi-signature schemes that are proven secure under the DDH assumption in the random oracle model (ROM). While the first one only achieves subtly weak unforgeability, the second one is as secure as the related schemes. We focus on the second one hereafter since this is exactly an improvement of the first one. Our scheme can ensure 128-bit security by implementing under the standardized elliptic curve (EC) P-384 due to the small reduction loss $O(Q_S)$, where Q_S is the number of signing queries of an adversary. Under provable secure parameters, the signature size and communication complexity of our scheme are the smallest among related schemes without using the AGM. Hence we can conclude that our scheme is an answer to our research question.

Practical Interpretation and Contribution of Our Result. For DL-based multi-signatures, we can consider three evaluation items: (i) reliability of the security (i.e., concrete security), (ii) efficiency, and (iii) strength of assumptions. Which multi-signature scheme is recommended depends on which of the items the user considers important for his application. For example, if the user requires a short signature size at the expense of others, MuSig2 and

DWMS are recommended since the sizes of their signatures are the shortest among all schemes under P-256. For users who are more concerned about the strength of assumptions than anything else, the recommended schemes are TZ and HBMS if he also wants the efficiency of online communication complexity and total communication complexity, respectively.

Our scheme is a new candidate for multi-signature schemes that can fulfill all three requirements. Indeed, our scheme is recommended for users who see these items as important since the security of our scheme is based on the DDH assumption and relies on only the ROM, and the signature size and total communication complexity are smallest among schemes without using the AGM. Using MuSig2-2, DWMS, HBMS-AGM, or LK allows us to ensure (i) and (ii) but it forces us to compromise (iii). We benefit from schemes without using the AGM, e.g., HBMS, TZ, PW-1, PW-2, with respect to (i) and (iii), but the signature sizes are about or larger than 2000-bit. While the signature size of MuSig2-1 is smaller than theirs, its security requires an interactive assumption. Thus we can conclude that our scheme is only one solution to fulfill all three requirements without compromising one of them. This result enhances the feasibility of cryptocurrencies and blockchain applications making them more suitable for the requirements of users.

For a fair discussion, we should note that our scheme has a limitation of (ii). Specifically, the first round of our signing protocol needs to be executed in the online phase. Thus it is difficult to recommend applying our scheme to applications in which online communication is much expensive. MuSig2-1, MuSig2-2, DWMS, and TZ are recommended for such applications. But we remind that we need to compromise some of the three requirements when we use them.

Theoretical Interpretation and Contribution of Our Result. When focusing solely on the scheme constructed by an approach, our scheme achieves the optimal signature size and communication complexity.[1] Currently, it is known that there are mainly two approaches known for constructing DL-based two-round multi-signature schemes. The first approach is the mBCJ-like approach in which a scheme is constructed by combining the signature scheme and the special commitment scheme. Our scheme and some related schemes, e.g., HBMS, HBMS-AGM, LK, mBCJ, PW-1, and PW-2, are constructed by following this approach. The second approach is the MuSig2-like approach in which a scheme is based on one-more style assumptions. MuSig2-1, MuSig2-2, DWMS, TZ are constructed by following this approach.

[1]Note that we ignore an optimization. Specifically, in PW-1 and PW-2, each signer sends to a seed used to generate the decommitment instead of the decommitment itself. This optimization can be applied to our scheme and mBCJ-like related schemes.

Before explaining why our scheme is optimal when focusing on the first approach, we roughly review this approach. In this approach, a two-round scheme is based on a signature scheme, like the Schnorr signature scheme, which is obtained by applying the Fiat-Shamir transform to the three-pass identification scheme, Σ-protocol [Cra96]. The size of the multi-signature and communication complexity depend on both the signature size of the based signature scheme and the sizes of the commitment and the decommitment of the commitment scheme. Moreover, due to using the special commitment scheme, at least $O(Q_S)$ reduction loss occurs.

Now we move to the analysis of the signature size and communication complexity of our scheme. Our scheme is built from the DDH-based tightly secure Katz-Wang signature scheme. This is one of the tightly secure signature schemes based on DL-type assumptions. In the signing protocol of our scheme, each signer sends two group elements and two scalars to all signers in the first and second rounds, respectively. For the first round, the number of group elements to be sent is equivalent to that of group elements of the first prover's message of the DDH-based lossy identification on which the Katz-Wang signature scheme is based. For the second round, one of the two scalars to be shared is equivalent to the second prover's message of the lossy identification and the other is the decommitment of the commitment scheme. For the details of the lossy identification, see Section 4.1.2. Our multi-signature consists of three scalars. Two of these scalars are *equivalent* to the signature of the Katz-Wang signature scheme while the remaining scalar is decommitment. Moreover, our scheme has only a reduction loss $O(Q_S)$. From the above analysis, as long as following this approach and adopting the Katz-Wang signature scheme, it is difficult to further reduce the signature size and communication complexity. Indeed, if the size of the multi-signature were reduced, then the decommitment would disappear or the signature size of the Katz-Wang signature scheme would be reduced to only one scalar. Such schemes are no longer based on the Katz-Wang signature scheme or a commitment scheme. Also, it is hard to use the property of the signature scheme or the special commitment scheme, i.e., lossiness and equivocability, if the commitment of the commitment scheme is one group element. Therefore we can conclude that our scheme achieves the optimal signature size and communication complexity when focusing on the first approach.

We now mention the drawback of our scheme, that the first round should be executed in the online phase, in the terms of theoretical aspect. This disadvantage arises from the use of the commitment scheme. In the mBCJ-like two-round scheme built from the first approach, the commitment keys need to be generated depending on the message and the aggregated key. This is to prevent the reuse of the commitment key. Remind that the reuse of the commitment key allows us to enable the ROS attack. The message to be signed and the set of signers are decided at the beginning of the online phase. Thus if we want to execute the first round in the offline phase, we require a way to generate the commitment keys that are distinct with overwhelming probability for each signing session. However, it seems to be infeasible since the signers are not necessarily synchronous and we are not allowed to communicate with each other and share some information before the offline phase to achieve a two-round protocol. Therefore, we need to attempt other approaches to overcome this obstacle, e.g., the MuSig2-like approach or a new approach.

Open Problems. Here we show two open problems for constructing a two-round multi-signature scheme with a small reduction loss.

The first one is whether or not we can construct such a scheme by following the MuSig2-like approach. We remind that this approach allows us to obtain a one-round scheme with pre-processing. This approach typically requires the one-more style assumptions. Specifically, MuSig2 and DWMS require the (algebraic) one-more DL problem and TZ requires the algebraic one-more preimage resistance [TZ23], which is established by the DL assumption. To ensure a small reduction loss without relying on the AGM, we require one-more style assumptions from which we can construct a signature scheme with tight security or a small reduction loss, e.g., the computational Diffie-Hellman (CDH) assumption and the DDH problem. Recently, Bacho et al. introduced one-more style assumptions that are established by the one-more CDH assumption or the standard DDH assumption. It is an interesting question if a two-round scheme with a small reduction loss can be constructed from such assumptions.

The second one is whether there exists another approach to construct two-round signature schemes. As described above, it is known that there are mainly two approaches. This fact can be observed when constructing the two-round scheme based on the signature scheme that is obtained by the Fiat-Shamir transform, e.g., lattice-based two-round multi-signatures [DOTT21, BTT22, Che23b]. Addressing this issue would expand the possibilities for constructing multi-signature schemes, not only DL-based schemes but also schemes based on other algebraic structures containing the post-quantum schemes.

Chapter 7

Conclusion

In this thesis, we attempt to construct an efficient two-round multi-signature scheme with a small reduction loss without using the AGM. As a result, we proposed two schemes as the answer to the research question. The first scheme achieves only slightly weak unforgeability, which corresponds to my publication in **Publication Related to This Thesis**. The second scheme achieves slightly strong unforgeability without compromising efficiency and supporting additional assumptions. This scheme is exactly an improvement of the first one. The security of our schemes was proven under the DDH assumption and the random oracle model. The reduction loss of both our schemes is small enough to guarantee the 128-bit security under the NIST standardized EC P-384. The signature size and total communication complexity of our schemes are the smallest among the related schemes without relying on the AGM under provable secure parameters. Moreover, by implementing our first scheme, we confirmed that our schemes have a realistic running time in practice.

Our result shows the new trade-off between reliability of security, efficiency, and strength of underlying assumptions. This provides a new candidate for multi-signatures that can fulfill the requirements of users. Our scheme is suitable for users who do not want to compromise on the reliability of the security, efficiency, and strength of assumptions. This leads to improved feasibility of cryptocurrencies and blockchain-based applications more suited to the user's requirements.

References

[AB21] Handan Kilinç Alper and Jeffrey Burdges. Two-round trip schnorr multi-signatures via delinearized witnesses. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 157–188, Virtual Event, August 2021. Springer, Heidelberg.

[AFLT12] Michel Abdalla, Pierre-Alain Fouque, Vadim Lyubashevsky, and Mehdi Tibouchi. Tightly-secure signatures from lossy identification schemes. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 572–590. Springer, Heidelberg, April 2012.

[AHK20] Thomas Agrikola, Dennis Hofheinz, and Julia Kastner. On instantiating the algebraic group model from falsifiable assumptions. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 96–126. Springer, Heidelberg, May 2020.

[AHY15] Nuttapong Attrapadung, Goichiro Hanaoka, and Shota Yamada. A framework for identity-based encryption with almost tight security. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 521–549. Springer, Heidelberg, November / December 2015.

[AM09] Jean-Philippe Aumasson and Willi Meier. Zero-sum distinguishers for reduced keccak-f and for the core functions of luffa and hamsi. *Presented at the rump session of CHES 2009*, 2009.

[BB04] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 56–73. Springer, Heidelberg, May 2004.

[BBB+18] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE Symposium on Security and Privacy*, pages 315–334. IEEE Computer Society Press, May 2018.

[BBM00] Mihir Bellare, Alexandra Boldyreva, and Silvio Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 259–274. Springer, Heidelberg, May 2000.

[BBP04] Mihir Bellare, Alexandra Boldyreva, and Adriana Palacio. An uninstantiable random-oracle-model scheme for a hybrid-encryption problem. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 171–188. Springer, Heidelberg, May 2004.

[BCJ08] Ali Bagherzandi, Jung Hee Cheon, and Stanislaw Jarecki. Multisignatures secure under the discrete logarithm assumption and a generalized forking lemma. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM CCS 2008*, pages 449–458. ACM Press, October 2008.

[BD21] Mihir Bellare and Wei Dai. Chain reductions for multi-signatures and the HBMS scheme. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part IV*, volume 13093 of *LNCS*, pages 650–678. Springer, Heidelberg, December 2021.

[BDN18] Dan Boneh, Manu Drijvers, and Gregory Neven. Compact multi-signatures for smaller blockchains. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 435–464. Springer, Heidelberg, December 2018.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer, Heidelberg, August 2001.

[BGLS03] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 416–432. Springer, Heidelberg, May 2003.

[BGOY07] Alexandra Boldyreva, Craig Gentry, Adam O'Neill, and Dae Hyun Yum. Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing. In Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors, *ACM CCS 2007*, pages 276–285. ACM Press, October 2007.

[BJ08] Ali Bagherzandi and Stanislaw Jarecki. Multisignatures using proofs of secret key possession, as secure as the Diffie-Hellman problem. In Rafail Ostrovsky, Roberto De Prisco, and Ivan Visconti, editors, *SCN 08*, volume 5229 of *LNCS*, pages 218–235. Springer, Heidelberg, September 2008.

[BJLS16] Christoph Bader, Tibor Jager, Yong Li, and Sven Schäge. On the impossibility of tight cryptographic reductions. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 273–304. Springer, Heidelberg, May 2016.

[BKKP15] Olivier Blazy, Saqib A. Kakvi, Eike Kiltz, and Jiaxin Pan. Tightly-secure signatures from chameleon hash functions. In Jonathan Katz, editor, *PKC 2015*, volume 9020 of *LNCS*, pages 256–279. Springer, Heidelberg, March / April 2015.

[BL16] Xavier Boyen and Qinyi Li. Towards tightly secure lattice short signature and id-based encryption. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 404–434. Springer, Heidelberg, December 2016.

[BLL+21] Fabrice Benhamouda, Tancrède Lepoint, Julian Loss, Michele Orrù, and Mariana Raykova. On the (in)security of ROS. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 33–53. Springer, Heidelberg, October 2021.

[BLS01] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 514–532. Springer, Heidelberg, December 2001.

[BN05] Mihir Bellare and Gregory Neven. New multi-signature schemes and a general forking lemma, 2005. https://soc1024.ece.illinois.edu/teaching/ece498ac/fall2018/forkinglemma.pdf.

[BN06] Mihir Bellare and Gregory Neven. Multi-signatures in the plain public-key model and a general forking lemma. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 390–399. ACM Press, October / November 2006.

[BNN07] Mihir Bellare, Chanathip Namprempre, and Gregory Neven. Unrestricted aggregate signatures. In Lars Arge, Christian Cachin, Tomasz Jurdzinski, and Andrzej Tarlecki, editors, *ICALP 2007*, volume 4596 of *LNCS*, pages 411–422. Springer, Heidelberg, July 2007.

[Bol03] Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In Yvo Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 31–46. Springer, Heidelberg, January 2003.

[BP04] Mihir Bellare and Adriana Palacio. The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 273–289. Springer, Heidelberg, August 2004.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *ACM CCS 93*, pages 62–73. ACM Press, November 1993.

[BR96] Mihir Bellare and Phillip Rogaway. The exact security of digital signatures: How to sign with RSA and Rabin. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 399–416. Springer, Heidelberg, May 1996.

[Bro10] Daniel RL Brown. Sec 2: Recommended elliptic curve domain parameters. *Standars for Efficient Cryptography*, 2010.

[BTT22] Cecilia Boschini, Akira Takahashi, and Mehdi Tibouchi. MuSig-L: Lattice-based multi-signature with single-round online phase. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part II*, volume 13508 of *LNCS*, pages 276–305. Springer, Heidelberg, August 2022.

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In *30th ACM STOC*, pages 209–218. ACM Press, May 1998.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. On the random-oracle methodology as applied to length-restricted signature schemes. In Moni Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 40–57. Springer, Heidelberg, February 2004.

[Che23a] Yanbo Chen. DualMS: Efficient lattice-based two-round multi-signature with trapdoor-free simulation. Cryptology ePrint Archive, Report 2023/263, 2023. https://eprint.iacr.org/2023/263.

[Che23b] Yanbo Chen. DualMS: Efficient lattice-based two-round multi-signature with trapdoor-free simulation. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part V*, volume 14085 of *LNCS*, pages 716–747. Springer, Heidelberg, August 2023.

[CKM+23] Elizabeth C. Crites, Chelsea Komlo, Mary Maller, Stefano Tessaro, and Chenzhi Zhu. Snowblind: A threshold blind signature in pairing-free groups. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part I*, volume 14081 of *LNCS*, pages 710–742. Springer, Heidelberg, August 2023.

[CMR+23] Lily Chen, Dustin Moody, Karen Randall, Andrew Regenscheid, and Angela Robinson. Recommendations for discrete logarithm-based cryptography: Elliptic curve domain parameters, 2023-02-02 05:02:00 2023.

[CMS12] Sanjit Chatterjee, Alfred Menezes, and Palash Sarkar. Another look at tightness. In Ali Miri and Serge Vaudenay, editors, *SAC 2011*, volume 7118 of *LNCS*, pages 293–319. Springer, Heidelberg, August 2012.

[Cor00] Jean-Sébastien Coron. On the exact security of full domain hash. In Mihir Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 229–235. Springer, Heidelberg, August 2000.

[Cra96] Ronald Cramer. *Modular design of secure, yet practical cryptographic protocols*. PhD thesis, University of Amsterdam, 1996.

[CS99] Ronald Cramer and Victor Shoup. Signature schemes based on the strong RSA assumption. In Juzar Motiwalla and Gene Tsudik, editors, *ACM CCS 99*, pages 46–51. ACM Press, November 1999.

[Dam92] Ivan Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 445–456. Springer, Heidelberg, August 1992.

[dAS20a] Monika di Angelo and Gernot Salzer. Wallet contracts on ethereum. In *IEEE ICBC*, pages 1–2. IEEE, 2020.

[dAS20b] Monika di Angelo and Gernot Salzer. Wallet contracts on ethereum – identification, types, usage, and profiles, 2020.

[DEF+19] Manu Drijvers, Kasra Edalatnejad, Bryan Ford, Eike Kiltz, Julian Loss, Gregory Neven, and Igors Stepanovs. On the security of two-round multi-signatures. In *2019 IEEE Symposium on Security and Privacy*, pages 1084–1101. IEEE Computer Society Press, May 2019.

[Des90] Yvo Desmedt. Abuses in cryptography and how to fight them. In Shafi Goldwasser, editor, *CRYPTO'88*, volume 403 of *LNCS*, pages 375–389. Springer, Heidelberg, August 1990.

[DF90] Yvo Desmedt and Yair Frankel. Threshold cryptosystems. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 307–315. Springer, Heidelberg, August 1990.

[DF92] Yvo Desmedt and Yair Frankel. Shared generation of authenticators and signatures (extended abstract). In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 457–469. Springer, Heidelberg, August 1992.

[DGJL21] Denis Diemert, Kai Gellert, Tibor Jager, and Lin Lyu. More efficient digital signatures with tight multi-user security. In Juan Garay, editor, *PKC 2021, Part II*, volume 12711 of *LNCS*, pages 1–31. Springer, Heidelberg, May 2021.

[DGNW20] Manu Drijvers, Sergey Gorbunov, Gregory Neven, and Hoeteck Wee. Pixel: Multi-signatures for consensus. In Srdjan Capkun and Franziska Roesner, editors, *USENIX Security 2020*, pages 2093–2110. USENIX Association, August 2020.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[DMO00] Hiroshi Doi, Masahiro Mambo, and Eiji Okamoto. On the security of the RSA-based multisignature scheme for various group structures. In Ed Dawson, Andrew Clark, and Colin Boyd, editors, *ACISP 00*, volume 1841 of *LNCS*, pages 352–367. Springer, Heidelberg, July 2000.

[DOP05] Yevgeniy Dodis, Roberto Oliveira, and Krzysztof Pietrzak. On the generic insecurity of the full domain hash. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 449–466. Springer, Heidelberg, August 2005.

[DOTT21] Ivan Damgård, Claudio Orlandi, Akira Takahashi, and Mehdi Tibouchi. Two-round n-out-of-n and multi-signatures and trapdoor commitment from lattices. In Juan Garay, editor, *PKC 2021, Part I*, volume 12710 of *LNCS*, pages 99–130. Springer, Heidelberg, May 2021.

[ES16] Rachid El Bansarkhani and Jan Sturm. An efficient lattice-based multisignature scheme with applications to bitcoins. In Sara Foresti and Giuseppe Persiano, editors, *CANS 16*, volume 10052 of *LNCS*, pages 140–155. Springer, Heidelberg, November 2016.

[FH19] Masayuki Fukumitsu and Shingo Hasegawa. A tightly-secure lattice-based multisignature. In *Proceedings of the 6th on ASIA Public-Key Cryptography Workshop*, APKC '19, page 3–11, New York, NY, USA, 2019. Association for Computing Machinery.

[FH20] Masayuki Fukumitsu and Shingo Hasegawa. A lattice-based provably secure multisignature scheme in quantum random oracle model. In Khoa Nguyen, Wenling Wu, Kwok-Yan Lam, and Huaxiong Wang, editors, *ProvSec 2020*, volume 12505 of *LNCS*, pages 45–64. Springer, Heidelberg, November / December 2020.

[FH21] Masayuki Fukumitsu and Shingo Hasegawa. A tightly secure ddh-based multisignature with public-key aggregation. *Int. J. Netw. Comput.*, 11(2):319–337, 2021.

[FKL18] Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 33–62. Springer, Heidelberg, August 2018.

[FPS20] Georg Fuchsbauer, Antoine Plouviez, and Yannick Seurin. Blind schnorr signatures and signed ElGamal encryption in the algebraic group model. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 63–95. Springer, Heidelberg, May 2020.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Heidelberg, August 1987.

[GCD+16] Junqing Gong, Jie Chen, Xiaolei Dong, Zhenfu Cao, and Shaohua Tang. Extended nested dual system groups, revisited. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part I*, volume 9614 of *LNCS*, pages 133–163. Springer, Heidelberg, March 2016.

[Gen06] Craig Gentry. Practical identity-based encryption without random oracles. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 445–464. Springer, Heidelberg, May / June 2006.

[GHKP18] Romain Gay, Dennis Hofheinz, Lisa Kohl, and Jiaxin Pan. More efficient (almost) tightly secure structure-preserving signatures. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 230–258. Springer, Heidelberg, April / May 2018.

[GHKW16] Romain Gay, Dennis Hofheinz, Eike Kiltz, and Hoeteck Wee. Tightly CCA-secure encryption without pairings. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 1–27. Springer, Heidelberg, May 2016.

[GHR99] Rosario Gennaro, Shai Halevi, and Tal Rabin. Secure hash-and-sign signatures without the random oracle. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 123–139. Springer, Heidelberg, May 1999.

[GJKW07] Eu-Jin Goh, Stanislaw Jarecki, Jonathan Katz, and Nan Wang. Efficient signature schemes with tight reductions to the Diffie-Hellman problems. *Journal of Cryptology*, 20(4):493–514, October 2007.

[GK03] Shafi Goldwasser and Yael Tauman Kalai. On the (in)security of the Fiat-Shamir paradigm. In *44th FOCS*, pages 102–115. IEEE Computer Society Press, October 2003.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, April 1988.

[GP10] Henri Gilbert and Thomas Peyrin. Super-sbox cryptanalysis: Improved attacks for AES-like permutations. In Seokhie Hong and Tetsu Iwata, editors, *FSE 2010*, volume 6147 of *LNCS*, pages 365–383. Springer, Heidelberg, February 2010.

[Gui20] Aurore Guillevic. A short-list of pairing-friendly curves resistant to special TNFS at the 128-bit security level. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part II*, volume 12111 of *LNCS*, pages 535–564. Springer, Heidelberg, May 2020.

[Har94] Lein Harn. Group-oriented (t, n) threshold digital signature scheme and digital multisignature. *IEE Proceedings-Computers and Digital Techniques*, 141(5):307–313, 1994.

[HJ12] Dennis Hofheinz and Tibor Jager. Tightly secure signatures and public-key encryption. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 590–607. Springer, Heidelberg, August 2012.

[HK89] L Harn and T Kiesler. New scheme for digital multisignatures. *Electronics letters*, 25(15):1002–1003, 1989.

[HKS15] Dennis Hofheinz, Jessica Koch, and Christoph Striecks. Identity-based encryption with (almost) tight security in the multi-instance, multi-ciphertext setting. In Jonathan Katz, editor, *PKC 2015*, volume 9020 of *LNCS*, pages 799–822. Springer, Heidelberg, March / April 2015.

[Hof17] Dennis Hofheinz. Adaptive partitioning. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part III*, volume 10212 of *LNCS*, pages 489–518. Springer, Heidelberg, April / May 2017.

[IN83] K. Itakura and K. Nakamura. A public-key cryptosystem suitable for digital multisignatures. *NEC research & development*, 1983.

[KB16] Taechan Kim and Razvan Barbulescu. Extended tower number field sieve: A new complexity for the medium prime case. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 543–571. Springer, Heidelberg, August 2016.

[KM15] Neal Koblitz and Alfred Menezes. The random oracle model: A twenty-year retrospective. Cryptology ePrint Archive, Report 2015/140, 2015. https://eprint.iacr.org/2015/140.

[KMP16] Eike Kiltz, Daniel Masny, and Jiaxin Pan. Optimal security proofs for signatures from identification schemes. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 33–61. Springer, Heidelberg, August 2016.

[KN10] Dmitry Khovratovich and Ivica Nikolic. Rotational cryptanalysis of ARX. In Seokhie Hong and Tetsu Iwata, editors, *FSE 2010*, volume 6147 of *LNCS*, pages 333–346. Springer, Heidelberg, February 2010.

[KP19] Julia Kastner and Jiaxin Pan. Towards instantiating the algebraic group model. Cryptology ePrint Archive, Report 2019/1018, 2019. https://eprint.iacr.org/2019/1018.

[KSH23] Rikuhiro Kojima, Jacob C. N. Schuldt, and Goichiro Hanaoka. A new pairing-based two-round tightly-secure multi-signature scheme with key aggregation. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, advpub:2023CIP0022, 2023.

[KZ20] Daniel Kales and Greg Zaverucha. An attack on some signature schemes constructed from five-pass identification schemes. In Stephan Krenn, Haya Shulman, and Serge Vaudenay, editors, *CANS 20*, volume 12579 of *LNCS*, pages 3–22. Springer, Heidelberg, December 2020.

[Lan96] Susan K. Langford. Weakness in some threshold cryptosystems. In Neal Koblitz, editor, *CRYPTO'96*, volume 1109 of *LNCS*, pages 74–82. Springer, Heidelberg, August 1996.

[LBG09] Duc-Phong Le, Alexis Bonnecaze, and Alban Gabillon. Multisignatures as secure as the Diffie-Hellman problem in the plain public-key model. In Hovav Shacham and Brent Waters, editors, *PAIRING 2009*, volume 5671 of *LNCS*, pages 35–51. Springer, Heidelberg, August 2009.

[LHL95] Chuan-Ming Li, Tzonelih Hwang, and Narn-Yih Lee. Threshold-multisignature schemes where suspected forgery implies traceability of adversarial shareholders. In Alfredo De Santis, editor, *EUROCRYPT'94*, volume 950 of *LNCS*, pages 194–204. Springer, Heidelberg, May 1995.

[LJYP14] Benoît Libert, Marc Joye, Moti Yung, and Thomas Peters. Concise multi-challenge CCA-secure encryption and signatures with almost tight security. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part II*, volume 8874 of *LNCS*, pages 1–21. Springer, Heidelberg, December 2014.

[LK22] Kwangsu Lee and Hyoseung Kim. Two-round multi-signatures from Okamoto signatures. Cryptology ePrint Archive, Report 2022/1117, 2022. https://eprint.iacr.org/2022/1117.

[LMR+09] Mario Lamberger, Florian Mendel, Christian Rechberger, Vincent Rijmen, and Martin Schläffer. Rebound distinguishers: Results on the full Whirlpool compression function. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 126–143. Springer, Heidelberg, December 2009.

[LOS+06] Steve Lu, Rafail Ostrovsky, Amit Sahai, Hovav Shacham, and Brent Waters. Sequential aggregate signatures and multisignatures without random oracles. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 465–485. Springer, Heidelberg, May / June 2006.

[LP20] Gaëtan Leurent and Thomas Peyrin. SHA-1 is a shambles: First chosen-prefix collision on SHA-1 and application to the PGP web of trust. In Srdjan Capkun and Franziska Roesner, editors, *USENIX Security 2020*, pages 1839–1856. USENIX Association, August 2020.

[LPJY15] Benoît Libert, Thomas Peters, Marc Joye, and Moti Yung. Compactly hiding linear spans - tightly secure constant-size simulation-sound QA-NIZK proofs and applications. In Tetsu

Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 681–707. Springer, Heidelberg, November / December 2015.

[Lyu09] Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616. Springer, Heidelberg, December 2009.

[Lyu12] Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 738–755. Springer, Heidelberg, April 2012.

[Mer78] Ralph C. Merkle. Secure communications over insecure channels. *Commun. ACM*, 21(4):294–299, apr 1978.

[MH78] R. Merkle and M. Hellman. Hiding information and signatures in trapdoor knapsacks. *IEEE Transactions on Information Theory*, 24(5):525–530, 1978.

[MH96] Markus Michels and Patrick Horster. On the risk of disruption in several multiparty signature schemes. In Kwangjo Kim and Tsutomu Matsumoto, editors, *ASIACRYPT'96*, volume 1163 of *LNCS*, pages 334–345. Springer, Heidelberg, November 1996.

[Mit22] Shigeo Mitsunari. mcl - a portable and fast pairing-based cryptography library, 2022. 2022/Apr/10 v1.60, https://github.com/herumi/mcl.

[MJ19] Changshe Ma and Mei Jiang. Practical lattice-based multisignature schemes for blockchains. *IEEE Access*, 7:179765–179778, 2019.

[MM00] Shirow Mitomi and Atsuko Miyaji. A multisignature scheme with message flexibility, order flexibility and order verifiability. In Ed Dawson, Andrew Clark, and Colin Boyd, editors, *ACISP 00*, volume 1841 of *LNCS*, pages 298–312. Springer, Heidelberg, July 2000.

[MOR01] Silvio Micali, Kazuo Ohta, and Leonid Reyzin. Accountable-subgroup multisignatures: Extended abstract. In Michael K. Reiter and Pierangela Samarati, editors, *ACM CCS 2001*, pages 245–254. ACM Press, November 2001.

[MPSW18] Gregory Maxwell, Andrew Poelstra, Yannick Seurin, and Pieter Wuille. Simple schnorr multi-signatures with applications to bitcoin. Cryptology ePrint Archive, Report 2018/068, 2018. https://eprint.iacr.org/2018/068.

[MPSW19] Gregory Maxwell, Andrew Poelstra, Yannick Seurin, and Pieter Wuille. Simple schnorr multi-signatures with applications to bitcoin. *Des. Codes Cryptogr.*, 87(9):2139–2164, 2019.

[MRH04] Ueli M. Maurer, Renato Renner, and Clemens Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In Moni Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 21–39. Springer, Heidelberg, February 2004.

[MWLD10] Changshe Ma, Jian Weng, Yingjiu Li, and Robert Deng. Efficient discrete logarithm based multi-signature scheme in the plain public key model. *Designs, Codes and Cryptography*, 54:121–133, 2010.

[Nak08] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008. https://bitcoin.org/bitcoin.pdf.

[Nat13] National institute of standards and technology. FIPS Pub 186-4 Federal Information Processing Standards Publication Digital Signature Standard (DSS). 2013.

[Nie02] Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 111–126. Springer, Heidelberg, August 2002.

[NRS21] Jonas Nick, Tim Ruffing, and Yannick Seurin. MuSig2: Simple two-round Schnorr multi-signatures. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 189–221, Virtual Event, August 2021. Springer, Heidelberg.

[NRSW20] Jonas Nick, Tim Ruffing, Yannick Seurin, and Pieter Wuille. MuSig-DN: Schnorr multi-signatures with verifiably deterministic nonces. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020*, pages 1717–1731. ACM Press, November 2020.

[Oka88] Tatsuaki Okamoto. A digital multisignature scheme using bijective public-key cryptosystems. *ACM Transactions on Computer Systems (TOCS)*, 6(4):432–441, 1988.

[Oka93] Tatsuaki Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In Ernest F. Brickell, editor, *CRYPTO'92*, volume 740 of *LNCS*, pages 31–53. Springer, Heidelberg, August 1993.

[OO93] Kazuo Ohta and Tatsuaki Okamoto. A digital multisignature scheme based on the Fiat-Shamir scheme. In Hideki Imai, Ronald L. Rivest, and Tsutomu Matsumoto, editors, *ASIACRYPT'91*, volume 739 of *LNCS*, pages 139–148. Springer, Heidelberg, November 1993.

[OO99] Kazuo Ohta and Tatsuaki Okamoto. Multi-signature schemes secure against active insider attacks. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 82(1):21–31, 1999.

[Ped92] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 129–140. Springer, Heidelberg, August 1992.

[Pol78] John M. Pollard. Monte Carlo methods for index computation mod p. *Mathematics of Computation*, 32:918–924, 1978.

[PP16] Jong Hwan Park and Young-Ho Park. A tightly-secure multisignature scheme with improved verification. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, 99-A(2):579–589, 2016.

[PPKW97] Sangjoon Park, Sangwoo Park, Kwangjo Kim, and Dongho Won. Two efficient RSA multisignature schemes. In Yongfei Han, Tatsuaki Okamoto, and Sihan Qing, editors, *ICICS 97*, volume 1334 of *LNCS*, pages 217–222. Springer, Heidelberg, November 1997.

[PS96] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 387–398. Springer, Heidelberg, May 1996.

[PS00] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, June 2000.

[PV05] Pascal Paillier and Damien Vergnaud. Discrete-log-based signatures may not be equivalent to discrete log. In Bimal K. Roy, editor, *ASIACRYPT 2005*, volume 3788 of *LNCS*, pages 1–20. Springer, Heidelberg, December 2005.

[PW22] Jiaxin Pan and Benedikt Wagner. Lattice-based signatures with tight adaptive corruptions and more. In Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe, editors, *PKC 2022, Part II*, volume 13178 of *LNCS*, pages 347–378. Springer, Heidelberg, March 2022.

[PW23] Jiaxin Pan and Benedikt Wagner. Chopsticks: Fork-free two-round multi-signatures from non-interactive assumptions. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 597–627. Springer, Heidelberg, April 2023.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the Association for Computing Machinery*, 21(2):120–126, February 1978.

[RSS11] Thomas Ristenpart, Hovav Shacham, and Thomas Shrimpton. Careful with composition: Limitations of the indifferentiability framework. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 487–506. Springer, Heidelberg, May 2011.

[RY07] Thomas Ristenpart and Scott Yilek. The power of proofs-of-possession: Securing multiparty signatures against rogue-key attacks. In Moni Naor, editor, *EUROCRYPT 2007*, volume 4515 of *LNCS*, pages 228–245. Springer, Heidelberg, May 2007.

[Sch90] Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 239–252. Springer, Heidelberg, August 1990.

[Sch11] Sven Schäge. Tight proofs for signature schemes without random oracles. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 189–206. Springer, Heidelberg, May 2011.

[SOK00] Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. Cryptosystems based on pairing. In *The 2000 Symposium on Cryptography and Information Security*, Japan, January 2000.

[SSH11] Koichi Sakumoto, Taizo Shirai, and Harunaga Hiwatari. Public-key identification schemes based on multivariate quadratic polynomials. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 706–723. Springer, Heidelberg, August 2011.

[STV+16] Ewa Syta, Iulia Tamas, Dylan Visher, David Isaac Wolinsky, Philipp Jovanovic, Linus Gasser, Nicolas Gailly, Ismail Khoffi, and Bryan Ford. Keeping authorities "honest or bust" with decentralized witness cosigning. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 526–545. Ieee, 2016.

[TZ23] Stefano Tessaro and Chenzhi Zhu. Threshold and multi-signature schemes from linear hash functions. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 628–658. Springer, Heidelberg, April 2023.

[Wag02] David Wagner. A generalized birthday problem. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 288–303. Springer, Heidelberg, August 2002.

[Wat05] Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 114–127. Springer, Heidelberg, May 2005.

[WS07] J. Wu and D.R. Stinson. An efficient identification protocol and the knowledge-of-exponent assumption. Cryptology ePrint Archive, Report 2007/479, 2007. https://eprint.iacr.org/2007/479.

[WSQL08] Zecheng Wang, Taozhi Si, Haifeng Qian, and Zhibin Li. A cdh-based multi-signature scheme with tight security reduction. In *Proceedings of the 9th International Conference for Young Computer Scientists, ICYCS 2008, Zhang Jia Jie, Hunan, China, November 18-21, 2008*, pages 2096–2101. IEEE Computer Society, 2008.

[Yan18] Naoto Yanai. Meeting tight security for multisignatures in the plain public key model. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, 101-A(9):1484–1493, 2018.

[Zha22a] Mark Zhandry. Augmented random oracles. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part III*, volume 13509 of *LNCS*, pages 35–65. Springer, Heidelberg, August 2022.

[Zha22b] Mark Zhandry. To label, or not to label (in generic groups). In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part III*, volume 13509 of *LNCS*, pages 66–96. Springer, Heidelberg, August 2022.

[ZZK22] Cong Zhang, Hong-Sheng Zhou, and Jonathan Katz. An analysis of the algebraic group model. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part IV*, volume 13794 of *LNCS*, pages 310–322. Springer, Heidelberg, December 2022.

List of Publications

Journal Paper

[1] Kaoru Takemure, Yusuke Sakai, Bagus Santoso, Goichiro Hanaoka, Kazuo Ohta, "More Efficient Two-Round Multi-Signature Scheme with Provably Secure Parameters for Standardized Elliptic Curve", IEICE Transaction on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E107-A, No.7, pp.-, Jul. 2024. (to appear, total number of pages: 24)

[2] Kaoru Takemure, Yusuke Sakai, Bagus Santoso, Goichiro Hanaoka, Kazuo Ohta, "Achieving Pairing-Free Aggregate Signatures using Pre-Communication between Signer", IEICE Transaction on Fundamentals of Electronics, Communications and Computer Sciences, 104-A(9): 1188-1205 (2021).

Refereed Conference Paper

[3] Kaoru Takemure, Yusuke Sakai, Bagus Santoso, Goichiro Hanaoka, Kazuo Ohta, "Achieving Pairing-Free Aggregate Signatures using Pre-Communication between Signers", Provable and Practical Security (ProvSec2020). Lecture Notes in Computer Science, vol 12505.

Preprint

[4] Goichiro Hanaoka, Kazuo Ohta, Yusuke Sakai, Bagus Santoso, Kaoru Takemure, Yunlei Zhao, "Cryptanalysis of Aggregate Γ-Signature and Practical Countermeasures in Application to Bitcoin", Cryptology ePrint Archive, Report2020/1484.

Non-Refereed Conference Papers and Posters

[5] 竹牟禮 薫, バグス サントソ,“任意の環におけるイデアル格子問題に基づいた本人確認方式”, 信学技報Vol.118, No.478, pp39-44, 2019.

[6] 竹牟禮 薫, バグス サントソ, 荒井 嵩博, “任意の環におけるイデアル格子問題に基づいた本人確認方式”, 2019年暗号と情報セキュリティシンポジウム(SCIS2019), 滋賀, 2019年1月.

[7] 竹牟禮 薫, 坂井 祐介, Bagus Santoso, 花岡 悟一郎, 太田 和夫, “事前通信モデルにおけるペアリングを用いない集約署名”, 2020年暗号と情報セキュリティシンポジウム(SCIS2020), 高知, 2020年1月.

[8] 竹牟禮 薫, 坂井 祐介, バグス サントソ, 花岡 悟一郎, 太田 和夫, “帰着ロスを考慮したパラメタの下でより効率的な2ラウンド多重署名方式”, 2023年暗号と情報セキュリティシンポジウム(SCIS2023), 福岡, 2023年1月.

[9] 横田 明卓, 竹牟禮 薫, Bagus Santoso, “新たなNP困難なMorphism of Polynomials問題に基づいた本人確認方式”, 2023年暗号と情報セキュリティシンポジウム(SCIS2023), 福岡, 2023年1月.

[10] Kaoru Takemure, Bagus Santoso, “Concurrently Secure Identification Schemes Based on the Hardness of Ideal Lattice Problems in all Rings and a General Simulatable Sampling”, 2019年情報理論とその応用シンポジウム(SITA2019), 鹿児島, 2019年11月.