IACR Communications in Cryptology
ISSN 3006-5496, Vol. 1, No. 1, 30 pages.

https://doi.org/10.62056/a6qj89n4e

An Explicit High-Moment Forking Lemma and its Applications to the Concrete Security of Multi-Signatures

Gil Segev and Liat Shapira

School of Computer Science and Engineering, Hebrew University of Jerusalem, Israel

Abstract. In this work we first present an explicit forking lemma that distills the information-theoretic essence of the high-moment technique introduced by Rotem and Segev (CRYPTO '21), who analyzed the security of identification protocols and Fiat-Shamir signature schemes. Whereas the technique of Rotem and Segev was particularly geared towards two specific cryptographic primitives, we present a stand-alone probabilistic lower bound, which does not involve any underlying primitive or idealized model. The key difference between our lemma and previous ones is that instead of focusing on the tradeoff between the *worst-case* or *expected* running time of the resulting forking algorithm and its success probability, we focus on the tradeoff between *higher moments* of its running time and its success probability.

Equipped with our lemma, we then establish concrete security bounds for the BN and BLS multi-signature schemes that are significantly tighter than the concrete security bounds established by Bellare and Neven (CCS '06) and Boneh, Drijvers and Neven (ASIACRYPT '18), respectively. Our analysis does not limit adversaries to any idealized algebraic model, such as the algebraic group model in which all algorithms are assumed to provide an algebraic justification for each group element they produce. Our bounds are derived in the random-oracle model based on the standard-model second-moment hardness of the discrete logarithm problem (for the BN scheme) and the computational co-Diffie-Hellman problem (for the BLS scheme). Such second-moment assumptions, asking that the success probability of any algorithm in solving the underlying computational problems is dominated by the second moment of the algorithm's running time, are particularly plausible in any group where no better-than-generic algorithms are currently known.

Keywords: Signature schemes · forking lemma · concrete security

1 Introduction

A multi-signature scheme [IN83, BN06] enables any set of signers, within a large and decentralized system, to jointly produce a compact signature on a given message. Research on the design and analysis of multi-signature schemes has recently gained significant renewed interest, as such schemes were found particularly suitable for blockchain applications (e.g., [BDN18, MPSW19]). This high level of suitability dates back to the work of Bellare and Neven [BN06], who showed that multi-signature schemes can be constructed in the *plain* public-key model. In this model, each signer locally produces their signing and verification keys, without engaging in an interactive key-generation process with other signers or with a

Supported by the Israel Science Foundation (Grant No. 1336/22) and by the European Union (ERC, FTRC, 101043243). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

E-mail: segev@cs.huji.ac.il (Gil Segev), liat.shapira@mail.huji.ac.il (Liat Shapira)

Received: 2024-01-01 Accepted: 2024-06-04

registration authority, and without augmenting verification keys with proofs of knowledge that need to be individually verified by all other signers.

Bellare and Neven constructed a multi-signature scheme (to which we refer as the BN multi-signature scheme), and established its security in the random-oracle model based on the hardness of the discrete logarithm (DL) problem in prime-order groups. A potential drawback, however, in some scenarios of the BN scheme is that its signing process requires interaction among the set of signers in the form of a three-round signing protocol. Motivated by blockchain applications, Boneh, Drijvers and Neven [BDN18] then showed that the BLS signature scheme [BLS01] can be extended to a multi-signature scheme (to which we refer as the BLS multi-signature scheme), whose signing process is non-interactive. Boneh, Drijvers and Neven established the security of the BLS multi-signature scheme in the random-oracle model based on the hardness of the computational co-Diffie-Hellman (co-CDH) problem in prime-order bilinear groups.

Following up on earlier constructions of multi-signature schemes in various different models (see [OO93, LHL95, MOR01, Bol03, LOS$^+$06, BGOY07, RY07] and the references therein), the renewed interest in such schemes has led to a host of new and exciting constructions in the plain public-key model, offering various trade-offs between their efficiency and security (e.g., [MPSW19, DEF$^+$19, NRSW20, AB21, BD21, NRS21, BTT22, DOTT22, FSZ22, LK23, PW23, TZ23]).

The concrete security of multi-signatures. The security of a wide variety of cryptographic schemes is established via the classic "forking lemma". The lemma, originally introduced in the seminal work of Pointcheval and Stern [PS00], and then generalized to a stand-alone probabilistic lower bound by Bellare and Neven [BN06] (see also [AABN02, BCC$^+$16, KMP16] and the references therein), has become a fundamental and extremely useful tool.

Specifically, for the BN scheme, the security proof presented by Bellare and Neven (see also the more refined analysis by Bellare and Dai [BD21]) relies on the forking lemma to transform any malicious forger that runs in time t, issues q_{H} random-oracle queries and breaks the security of the scheme with probability ϵ, into a DL algorithm that runs in time roughly t and has success probability roughly $\epsilon^2/q_{\mathsf{H}}$ (in Section 4 we provide a formal statement of their result). Thus, in any group of order p in which Shoup's generic hardness result for computing discrete logarithms is believed to hold [Sho97][1], this leads to the concrete bound $\epsilon \leq (q_{\mathsf{H}} \cdot t^2/p)^{1/2}$ on the security of the BN scheme.[2] However, there are currently no known attacks on the BN scheme that are better than computing discrete logarithms, for which the best-known algorithms offer a success probability of only t^2/p. This substantial "square-root" gap, especially for 256-bit groups, arises in the analysis of a variety of cryptographic schemes that rely on the forking lemma (see [BD20, JT20, RS21] for in-depth discussions). Moreover, more recent multi-signature schemes in the DL-setting (e.g., [MPSW19, BD21]), whose known security proofs rely on *nested* applications of the forking lemma, exhibit larger gaps.

For the BLS scheme, the security proof presented by Boneh, Drijvers and Neven [BDN18] exhibits an even more significant gap. Specifically, Boneh, Drijvers and Neven relied on the forking lemma to transform any malicious forger that runs in time t, issues q_{H} random-oracle queries, and breaks the security of the scheme with probability ϵ, into a co-CDH algorithm that runs in time roughly $q_{\mathsf{H}}^2 \cdot t/\epsilon$ and has success probability roughly ϵ/q_{H} (in Section 5 we provide a formal statement of their result). Thus, in bilinear groups of prime order p in which one assumes that the co-CDH problem is as hard as in the

[1] That is, that any algorithm running in time T solves the DL problem with probability at most T^2/p.

[2] Other formulations of the forking lemma (e.g., [PS00, AABN02, BCC$^+$16, KMP16]) lead to various similar trade-offs between the success probability and the running time of the resulting discrete-logarithm algorithm. However, as discussed by Bellare and Dai [BD20] and by Jaeger and Tessaro [JT20], they all face the same square-root loss.

generic bilinear-group model[3], this leads to the concrete bound $\epsilon \leq (q_{\mathsf{H}}^5 \cdot t^2/p)^{1/3}$ on the security of the BLS multi-signature scheme. For various realistic ranges of the malicious forger's running time t and number of queries q_{H}, this bound may fall somewhat short of providing sufficient guarantees (once again, especially for 256-bit groups).

Tighter concrete security within the algebraic group model. The above-described significant gaps in the concrete security of multi-signature schemes have so far been addressed mainly by proving security with respect to restricted classes of attackers. Specifically, tighter concrete security bounds were established for multi-signature schemes [AB21, BD21, NRS21, LK23] with respect to algebraic attackers within the idealized algebraic group model [FKL18]. In this idealized model, all algorithms are assumed to provide an algebraic justification for each group element that they produce. Such an algebraic justification typically enables to prove the security of schemes without relying on the forking lemma and thus avoids the resulting security loss [AHK20, BFL20, FPS20, MTT19, RS20].[4]

This approach is significantly refined by the work of Bellare and Dai [BD20, BD21], who introduced the following two-step modular analysis: (1) Given a (single-signer or multi-signer) signature scheme, identify a (possibly interactive) computational problem whose concrete hardness is tightly equivalent to the concrete security of the scheme, but which can be described in a more direct and elegant manner, and then (2) derive concrete security bounds by reducing this problem to the DL problem. However, using the forking lemma for these reductions then leads to the above-discussed concrete security gaps. Therefore, to avoid these gaps, the algebraic group model is nevertheless utilized.

Tighter concrete security without the algebraic group model? Rotem and Segev [RS21] provided tighter concrete security bounds for Σ-protocols and their associated Fiat-Shamir signature schemes [FS87, AABN02]. Instead of analyzing security in an idealized algebraic model, they introduced a different forking technique, relying on the assumption that the underlying computational problem is "d-th moment hard": The success probability of any algorithm in solving it is dominated by the d-th moment of the algorithm's running time. Equipped with such an underlying assumption, their forking technique transforms a malicious attacker into an algorithm solving the underlying computational problem by optimizing the trade-off between the algorithm's success probability and the d-th moment of their running time.

In the concrete context of the DL problem, the assumption that the problem is d-moment hard states that any algorithm running in time T solves the DL problem with probability at most $\mathbb{E}[T^d]/p$, where $\mathbb{E}[T^d]$ denotes the d-th moment of the distribution corresponding to the algorithm's running time (see Section 2.1 for the formal definition of d-moment hardness). Shoup's original proof shows that the DL problem is 2-moment hard in the generic-group model [Sho97], and thus the second-moment DL assumption can be viewed as a highly plausible strengthening of the DL assumption in any group where no better-than-generic algorithms are currently known. More generally, the recent work of Segev, Sharabi and Yogev [SSY23] provided a generic framework for analyzing the d-moment hardness of a wide range of computational problems (refining and extending the work of Jeager and Tessaro [JT20] on expected-time hardness).

The forking technique introduced by Rotem and Segev, however, was particularly geared towards analyzing the security of Σ-protocols and their associated Fiat-Shamir (single-signer) signature schemes. In contrast to the work of Bellare and Neven [BN06], they did not explicitly provide a stand-along probabilistic tool, or any other indication of

[3] That is, that any algorithm running in time T solves the co-CDH problem with probability at most T^2/p [Sho97, Mau05, BB08].

[4] For security proofs that consist of nested applications of the forking lemma, the algebraic justification assumed to be provided by algorithms in the idealized algebraic group model enables to reduce the nesting depth, and thus to reduce the resulting security loss.

the extent to which their approach may be applicable for other cryptographic purposes – such as obtaining tighter concrete security bounds for multi-signature schemes.

1.1 Our Contributions

In this work we first present an explicit high-moment forking lemma that distills the information-theoretic essence of the technique introduced by Rotem and Segev [RS21]. Similarly to the general forking lemma of Bellare and Neven [BN06], our high-moment generalization consists of a stand-alone probabilistic lower bound, which does not involve any underlying cryptographic primitive (such as a signature scheme) or any idealized model (such as the random-oracle model). At a very high level, the key difference between our approach and that of Bellare and Neven is that whereas their forking lemma may be viewed as optimizing the tradeoff between the *worst-case* running time of the resulting forking algorithm and its success probability, our lemma focuses on optimizing the tradeoff between the *d-th moment* of its running time and its success probability.

Then, equipped with our lemma, we establish concrete security bounds for the BN and BLS multi-signature schemes that are tighter than the concrete security bounds established by Bellare and Neven [BN06] and Boneh, Drijvers and Neven [BDN18], respectively. Our tighter bounds are derived in the random-oracle model based on the standard-model second-moment hardness of the discrete logarithm problem (for the BN scheme) and the computational co-Diffie-Hellman problem (for the BLS scheme). We prove the following theorems (which, for simplicity, are stated here rather informally[5]):

Theorem 1 (informal)**.** *Let $\mathbb{G}$ be a cyclic group of prime order p. Assuming that the DL problem is second-moment hard in $\mathbb{G}$, then for any adversary that runs in time t, issues q_{H} random oracle queries, and breaks the security of the BN multi-signature scheme with probability ϵ, it holds that*

$$\epsilon \leq \left(q_{\mathsf{H}} \cdot \frac{t^2}{p} \right)^{2/3} .$$

Theorem 2 (informal)**.** *Let $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t)$ be a triplet of cyclic groups of prime order p equipped with a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$. Assuming that co-CDH problem is second-moment hard in $\mathcal{G}$, then for any adversary that runs in time t, issues q_{H} random oracle queries, and breaks the security of the BLS multi-signature scheme with probability ϵ, it holds that*

$$\epsilon \leq \left(q_{\mathsf{H}}^{5/2} \cdot \frac{t^2}{p} \right)^{2/3} .$$

Recall, as discussed above, that the analysis of Bellare and Neven provided the bound $\epsilon \leq (q_{\mathsf{H}} \cdot t^2/p)^{1/2}$, and the analysis of Boneh, Drijvers and Neven provided the bound $\epsilon \leq (q_{\mathsf{H}}^5 \cdot t^2/p)^{1/3}$. Thus, compared to their bounds, our bounds are significantly tighter as we increase the exponent from 1/2 to 2/3 for the BN scheme, and from 1/3 to 2/3 for the BLS scheme. For example, from the practical perspective of a 256-bit group, the security bounds established by Bellare and Neven and by Boneh, Drijvers and Neven show that any attacker that runs in time at most $t = 2^{64}$ and issues at most $q = 2^{30}$ random oracle queries breaks the BN multi-signature scheme with probability at most 2^{-49} and the BLS multi-signature scheme with probability at most 1. Our tighter bounds improve these to 2^{-65} and 2^{-35}, respectively. Although these bounds still do not match the best-possible "generic group" bound, we believe they provide a significant step towards better understanding these schemes without relying on idealized algebraic models. Tables 1 and 2 below provide additional such concrete examples.

[5] Most notably, these two statements do not include the number of signing queries issued by the adversary and various other lower-order terms.

Table 1: A comparison of the concrete security guarantees for the Bellare–Neven multi-signature scheme in the standard model.

Attacker's running time t	**Security parameter** p	**Oracle Queries** q	**Previous bound** $\left(q \cdot \frac{t^2}{p}\right)^{\frac{1}{2}}$	**Our bound** $\left(q \cdot \frac{t^2}{p}\right)^{\frac{2}{3}}$
2^{64}	2^{256}	2^{25}	$2^{-51.5}$	$2^{-68.67}$
2^{64}	2^{256}	2^{30}	2^{-49}	$2^{-65.33}$
2^{80}	2^{256}	2^{30}	2^{-33}	2^{-44}
2^{80}	2^{512}	2^{30}	2^{-161}	$2^{-214.67}$
2^{100}	2^{512}	2^{40}	2^{-136}	$2^{-181.33}$

Table 2: A comparison of the concrete security guarantees for the BLS multi-signature scheme in the standard model.

Attacker's running time t	**Security parameter** p	**Oracle Queries** q_H	**Previous bound** $\left(q_H^5 \cdot \frac{t^2}{p}\right)^{\frac{1}{3}}$	**Our bound** $\left(q_H^{\frac{5}{2}} \cdot \frac{t^2}{p}\right)^{\frac{2}{3}}$
2^{64}	2^{256}	2^{25}	2^{-1}	$2^{-43.67}$
2^{64}	2^{256}	2^{30}	> 1	$2^{-35.33}$
2^{70}	2^{512}	2^{25}	$2^{-82.33}$	$2^{-206.33}$
2^{80}	2^{512}	2^{30}	$2^{-67.33}$	$2^{-184.67}$
2^{100}	2^{512}	2^{40}	$2^{-37.33}$	$2^{-141.33}$

1.2 Paper Organization

The remainder of this paper is organized as follows. First, in Section 2 we present the basic notions of d-th moment hardness and multi-signature schemes. In Section 3 we formalize and prove an explicit high-moment forking lemma. In Sections 4 and 5 we then rely on our lemma for establishing tighter concrete security bounds for the BN and BLS multi-signature schemes, respectively.

2 Preliminaries

For an integer $n \in \mathbb{N}$ we denote by $[n]$ the set $\{1, \ldots, n\}$. For a distribution X we denote by $x \leftarrow X$ the process of sampling a value x from the distribution X. Similarly, for a set $\mathcal{X}$ we denote by $x \leftarrow \mathcal{X}$ the process of sampling a value x from the uniform distribution over $\mathcal{X}$. In the remainder of this section, we present the notion of d-th moment hardness and the standard notion of security for multi-signature schemes.

2.1 d-th Moment Hardness

We consider relations $\mathcal{R} = \{\mathcal{R}_\lambda\}_{\lambda \in \mathbb{N}}$, where $\mathcal{R}_\lambda \subseteq \mathcal{X}_\lambda \times \mathcal{W}_\lambda$ for any $\lambda \in \mathbb{N}$, and distributions $\mathcal{D} = \{\mathcal{D}_\lambda\}_{\lambda \in \mathbb{N}}$ where each $\mathcal{D}_\lambda$ produces pairs $(x, w) \in \mathcal{R}_\lambda$. For any probabilistic algorithm A and for any input $x \in \{0,1\}^*$ we denote by $\mathsf{Time}(A(x))$ the random variable corresponding to the running time of the computation $A(x)$ over the internal randomness of A.

Definition 1 ([RS21]). Let $d = d(\lambda)$, $\Delta = \Delta(\lambda)$ and $\delta = \delta(\lambda)$ be functions of the security parameter $\lambda \in \mathbb{N}$, and let $\mathcal{R} = \{\mathcal{R}_\lambda\}_{\lambda \in \mathbb{N}}$ be a relation, where $\mathcal{R}_\lambda \subseteq \mathcal{X}_\lambda \times \mathcal{W}_\lambda$ for any $\lambda \in \mathbb{N}$.

We say that $\mathcal{R}$ is *d-moment (Δ,δ)-hard with respect to a distribution* $\mathcal{D} = \{\mathcal{D}_\lambda\}_{\lambda\in\mathbb{N}}$ if for every algorithm A it holds that

$$\Pr_{(x,w)\leftarrow\mathcal{D}_\lambda}[(x,A(x))\in\mathcal{R}_\lambda] \le \frac{\Delta\cdot\mathbb{E}_{(x,w)\leftarrow\mathcal{D}_\lambda}\left[(\mathsf{Time}(A(x)))^d\right]}{|\mathcal{W}_\lambda|^\delta},$$

for all sufficiently large $\lambda\in\mathbb{N}$, where the probability is additionally taken over the internal randomness of A.

For the computational problems considered in this work (as discussed in the remainder of this section), the known generic-group hardness results show that it suffices to consider the above definition when setting $\Delta(\lambda)=1$ and $\delta(\lambda)=1$ for all $\lambda\in\mathbb{N}$. For this setting of the parameters, we will simply say that a relation $\mathcal{R}$ is d-moment hard instead of d-moment $(1,1)$-hard. When stating our results, we thus consider the setting of $\Delta(\lambda)=1$ and $\delta(\lambda)=1$, and note that all of our results in fact hold for any setting of Δ and δ.

The DL and co-CDH relations. In Section 4, for proving a tighter concrete security bound for the BN scheme [BN06], we consider the above definition in the case of the discrete logarithm problem. In this case, the underlying distribution $\mathcal{D}$ and relation $\mathcal{R}$ are defined as follows:

- The distribution $\mathcal{D}_\lambda$ first invokes a group-generation algorithm $\mathsf{GroupGen}(1^\lambda)$ for producing the description $(\mathbb{G},p,g)$ of a cyclic group of order q that is generated by g, where p is a λ-bit prime. Then, it uniformly samples $h\leftarrow\mathbb{G}$ and lets $x=(\mathbb{G},p,g,h)$.
- The relation $\mathcal{R}$ consists of all such pairs $((\mathbb{G},p,g,h),w)$ where $h=g^w$ for $w\in\mathbb{Z}_p$.

As discussed by Rotem and Segev [RS21], given that the discrete logarithm problem is second-moment hard in the generic-group model [Sho97, JT20, SSY23], the assumption that the discrete logarithm (DL) problem is second-moment hard in the standard model can be viewed as capturing the problem's generic hardness in the form of a standard-model assumption.

Similarly, for proving a tighter concrete security bound for the BLS multi-signature scheme due to Boneh, Drijvers and Neven [BDN18], we consider the above definition in the case of the computational co-Diffie-Hellman problem (co-CDH). In this case, the underlying distribution $\mathcal{D}$ and relation $\mathcal{R}$ are defined as follows:

- The distribution $\mathcal{D}$ first invokes a group-generation algorithm $\mathsf{GroupGen}(1^\lambda)$ for producing the description $(\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_t,p,g_1,g_2,e)$ of three cyclic groups of order p, where $\mathbb{G}_1$ is generated by g_1, $\mathbb{G}_2$ is generated by g_2, p is a λ-bit prime, and $e:\mathbb{G}_1\times\mathbb{G}_2\to\mathbb{G}_t$ is an efficiently-computable non-degenerated bilinear map (in fact, the group $\mathbb{G}_t$ and the bilinear map e are not essential in order to define the distribution $\mathcal{D}$ and the relation $\mathcal{R}$). Then, it uniformly samples $\alpha,\beta\leftarrow\mathbb{Z}_p$ and lets $x=\left(\mathbb{G}_1,\mathbb{G}_2,q,g_1,g_2,g_1^\alpha,g_2^\alpha,g_2^\beta\right)$.
- The relation $\mathcal{R}$ consists of all pairs $\left(\left(\mathbb{G}_1,\mathbb{G}_2,p,g_1,g_2,g_1^\alpha,g_2^\alpha,g_2^\beta\right),w\right)$ where $w=g_1^{\alpha\beta}\in\mathbb{G}_1$ for $\alpha,\beta\in\mathbb{Z}_p$.

The recent work of Segev, Sharabi and Yogev [SSY23] provided a generic framework for analyzing the d-moment hardness of a wide range of computational problems (refining and extending the work of Jeager and Tessaro [JT20] on expected-time hardness). Their framework, together with the classic generic hardness results for the computational Diffie-Hellman problem [Sho97], establish the second-moment hardness of the computational co-Diffie-Hellman problem in the generic-group model.

2.2 Multi-Signature Schemes

A multi-signature scheme [IN83, BN06] is a six-tuple $\Pi = (\mathsf{Setup}, \mathsf{KG}, \mathsf{KeyAgg}, \mathsf{Sign}, \mathsf{SigAgg}, \mathsf{Verify})$ of polynomial-time algorithms. The setup algorithm Setup receives as input the unary representation of the security parameter $\lambda \in \mathbb{N}$ and outputs public parameters pp. The key-generation algorithm KG receives as input the public parameters pp, and outputs a signing key sk and a verification key vk. The key-aggregation algorithm KeyAgg is a deterministic algorithm that takes as input the public parameters pp and a vector of verification keys $\vec{\mathsf{vk}}$, and outputs an aggregated verification key aggvk.[6] It should be noted that not all multi-signature schemes offer a non-trivial key-aggregation algorithm. In such cases (e.g., the BN scheme [BN06]), we can view its key-aggregation algorithm as the identity function that takes as input a vector $\vec{\mathsf{vk}}$ of verification keys and outputs it as the aggregated key aggvk.

For schemes with non-interactive signing, the signing algorithm Sign receives as input the public parameters pp, a signing key sk, a vector $\vec{\mathsf{vk}}$ of verification keys, and a message m that is taken from a message space $\mathcal{M}$, and outputs a signature σ. For schemes with interactive signing, the signing algorithm defines an interactive protocol by additionally receiving as input at each round the relevant party's internal state and the communication produced by all other parties. The signature-aggregation algorithm SigAgg is a deterministic algorithm that takes as input the public parameters pp, a vector of verification keys $\vec{\mathsf{vk}}$, and a vector of signatures $\vec{\sigma}$, and outputs an aggregated signature σ. Finally, the verification algorithm Verify receives as input the public parameters pp, an aggregated verification key aggvk, a message m and an aggregated signature σ, and outputs either 0 or 1.

In terms of correctness, we consider the following requirement, which we formalize for simplicity for schemes with non-interactive signing and without random oracles. We then discuss its standard extensions to consider interactive signing and random oracles.

Definition 2. A multi-signature scheme $\Pi = (\mathsf{Setup}, \mathsf{KG}, \mathsf{KeyAgg}, \mathsf{Sign}, \mathsf{SigAgg}, \mathsf{Verify})$ with non-interactive signing over a message space $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda \in \mathbb{N}}$ is *correct* if there exists a negligible function $\nu = \nu(\cdot)$ such that for any polynomial number $n = n(\cdot)$ of signers, security parameter $\lambda \in \mathbb{N}$, and message $m \in \mathcal{M}_\lambda$ it holds that

$$\Pr\left[\mathsf{Verify}\big(\mathsf{pp}, \mathsf{KeyAgg}\big(\mathsf{pp}, \vec{\mathsf{vk}}\big), m, \mathsf{SigAgg}\big(\mathsf{pp}, \vec{\mathsf{vk}}, \vec{\sigma}\big)\big) = 1\right] \geq 1 - \nu(\lambda)$$

for $\vec{\mathsf{vk}} = (\mathsf{vk}_1, \ldots, \mathsf{vk}_n)$ and $\vec{\sigma} = (\sigma_1, \ldots, \sigma_n)$, where the probability is taken over the choice of $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, and over the choices of $(\mathsf{sk}_i, \mathsf{vk}_i) \leftarrow \mathsf{KG}(\mathsf{pp})$ and $\sigma_i \leftarrow \mathsf{Sign}\big(\mathsf{pp}, \mathsf{sk}_i, \vec{\mathsf{vk}}, m\big)$ for every $i \in [n]$.

As noted above, Definition 2 may be extended in various manners. These include, in particular, the following two standard extensions:

- Interactive signing: Definition 2 extends to schemes with interactive signing by letting $(\sigma_1, \ldots, \sigma_n)$ denote the local output of each party in the interactive signing protocol, where each party $i \in [n]$ is provided with $\big(\mathsf{pp}, \mathsf{sk}_i, \vec{\mathsf{vk}}, m\big)$ as input. We refer the reader to the work of Bellare and Dai [BD21] for a formal treatment of the correctness requirement for schemes with interactive signing.

- Random-oracle model: Definition 2 extends to schemes whose security is analyzed in the random-oracle model [BR93] by augmenting all algorithms with access to the random oracle, and considering all probabilities also over the randomness of the oracle.

[6] For concreteness, we view collections of verification keys as vectors and not sets, noting that any set can be uniquely transformed into a vector by determining an ordering among its elements (e.g., lexicographic order).

In terms of security, the standard notion of security for multi-signature schemes [BN06] considers adversaries that obtain a single honestly-generated verification key vk, and can then adaptively issue any polynomial number of signing queries. Each such query consists of a message m and a set of signers in the form of a vector $\vec{\mathsf{vk}}$ of verification keys that contains the honestly-generated verification key vk. The goal of such an adversary is to output a triplet $\left(\vec{\mathsf{vk}}^*, m^*, \mathsf{agg}\sigma^*\right)$, where: (1) $\vec{\mathsf{vk}}^*$ contains the honestly-generated verification key vk, (2) the adversary did not issue a signing query for $\left(\vec{\mathsf{vk}}^*, m^*\right)$, and (3) $\mathsf{agg}\sigma^*$ is a valid aggregated signature on the message m^* with respect to $\vec{\mathsf{vk}}^*$. This is captured by the following definition, which we again formalize for simplicity for schemes with non-interactive signing and without random oracles, and then discuss its standard extensions to consider interactive signing and random oracles.

Definition 3. Let $t = t(\lambda), q_{\mathsf{sign}} = q_{\mathsf{sign}}(\lambda)$, and $\epsilon = \epsilon(\lambda)$ be functions of the security parameter $\lambda \in \mathbb{N}$. A multi-signature scheme $\Pi = (\mathsf{Setup}, \mathsf{KG}, \mathsf{KeyAgg}, \mathsf{Sign}, \mathsf{SigAgg}, \mathsf{Verify})$ with non-interactive signing is $(t, q_{\mathsf{sign}}, \epsilon)$*-unforgeable* if for any algorithm A that runs in time at most t and issues at most q_{sign} signing queries, it holds that

$$\mathsf{Adv}_{\Pi}^{\mathsf{MultiSig}}(A, \lambda) \overset{\mathsf{def}}{=} \Pr\left[\mathsf{Exp}_{\Pi}^{\mathsf{MultiSig}}(A, \lambda) = 1\right] \leq \epsilon(\lambda)$$

for all sufficiently large $\lambda \in \mathbb{N}$, where the experiment $\mathsf{Exp}_{\Pi}^{\mathsf{MultiSig}}(\mathcal{A}, \lambda)$ is defined as follows:

1. $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$.
2. $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KG}(\mathsf{pp})$.
3. $\left(\vec{\mathsf{vk}}^*, m^*, \mathsf{agg}\sigma^*\right) \leftarrow A^{\mathsf{Sign}(\mathsf{pp},\mathsf{sk},\cdot,\cdot)}(1^\lambda, \mathsf{pp}, \mathsf{vk})$.
4. If the following conditions are satisfied then output 1 and otherwise output 0:
   (a) $\mathsf{vk} \in \vec{\mathsf{vk}}^*$.
   (b) A did not query the oracle $\mathsf{Sign}(\mathsf{pp}, \mathsf{sk}, \cdot, \cdot)$ with $\left(\vec{\mathsf{vk}}^*, m^*\right)$.
   (c) $\mathsf{Verify}\left(\mathsf{pp}, \mathsf{KeyAgg}\left(\vec{\mathsf{vk}}^*\right), m^*, \mathsf{agg}\sigma^*\right) = 1$.

As discussed above, Definition 3 naturally extends to consider interactive signing and random oracles:

- Interactive signing: Definition 3 extends to schemes with an interactive signing protocol by providing adversaries with access to a stateful signing oracle [BN06, BD21]. This stateful oracle enables the initiation of new signing sessions and the execution of previously initiated ones in an adversarial manner. For our work, for the case of the BN scheme (whose signing is interactive), we do not directly analyze the security of the scheme and, therefore, do not require a formal extension of Definition 3 to schemes with interactive signing. Instead, we analyze the hardness of an interactive computational problem, which Bellare and Dai [BD21] proved to imply the security of their scheme. For the case of the BLS multi-signature scheme [BDN18], which we do analyze directly, Definition 3 suffices as the scheme has non-interactive signing.

- Random-oracle model: Definition 3 extends to schemes whose security is analyzed in the random-oracle model [BR93] by augmenting all algorithms (including the adversary) with access to the random oracle, introducing an additional parameter q_{H} that upper bounds the number of direct random-oracle queries issued by the adversary, and considering all probabilities also over the randomness of the oracle.

A relaxed notion of unforgeability. The goal of the adversary in the experiment $\mathsf{Exp}_{\Pi}^{\mathsf{MultiSig}}$ described in Definition 3 is to output a valid forgery $\left(\vec{\mathsf{vk}}^*, m^*, \mathsf{agg}\sigma^*\right)$ where: (1) $\vec{\mathsf{vk}}^*$ contains the honestly-generated verification key vk, and (2) the adversary did not issue a signing query for $\left(\vec{\mathsf{vk}}^*, m^*\right)$. However, Boneh and Drijvers and Neven [BDN18] proved the security of the BLS multi-signature scheme with respect to a more relaxed notion, in which the adversary is not allowed to issue *any* signing query involving the message m^*. Formally, we denote by $\mathsf{Exp}_{\Pi}^{\mathsf{rMultiSig}}$ the experiment corresponding to this relaxed notion, which is obtained from the experiment $\mathsf{Exp}_{\Pi}^{\mathsf{MultiSig}}$ by replacing Item 4b with the requirement that A did not query the oracle $\mathsf{Sign}(\mathsf{pp}, \mathsf{sk}, \cdot, \cdot)$ with $\left(\vec{\mathsf{vk}}, m^*\right)$ for any vector $\vec{\mathsf{vk}}$ of verification keys.

On the one hand, it should be noted that the BLS scheme is, in fact, insecure with respect to the more standard notion. However, on the other hand, it can be easily modified into one that satisfies the more standard notion: Instead of signing a message m with respect to $\vec{\mathsf{vk}}$, sign the message $(\vec{\mathsf{vk}}, m)$ with respect to $\vec{\mathsf{vk}}$. This simple modification does not introduce a significant overhead since the signed message is anyway first hashed via a random oracle, and essentially the exact same proof of Boneh and Drijvers and Neven goes through. However, to enable a direct-as-possible comparison to the concrete security bound proved by Boneh and Drijvers and Neven [BDN18], we analyze the concrete security of the BLS scheme with respect to this relaxed notion.

Asymptotic vs. fixed-group analysis. The above discussion of the security of multi-signature schemes, as well as the discussion of the DL and co-CDH problems in Section 2.1, are of an asymptotic flavor. Specifically, they consider an explicit security parameter $\lambda \in \mathbb{N}$ which is given as input to a setup algorithm $\mathsf{Setup}(1^\lambda)$ or group-generation algorithm $\mathsf{GroupGen}(1^\lambda)$ (and thus indirectly also to all other algorithms), and provide guarantees for all sufficiently large $\lambda \in \mathbb{N}$.

When analyzing the security of the BN and BLS multi-signature schemes in Sections 4 and 5, respectively, we follow a more concrete approach, in which a description of an underlying group is fixed in advance (its prime order p can essentially be viewed as a concrete analogous security parameter). It is important to note that we do not rely on any assumed properties related to the structure of the group (beyond its prime order) or to the representation of its elements. This standard approach enables to provide concrete security guarantees for any fixed-size candidate group in which the underlying cryptographic problems (e.g., DL or co-CDH) are assumed to be computationally hard (in our case, d-moment hard).

3 An Explicit High-Moment Forking Lemma

In this section we present an explicit high-moment forking lemma that distills the information-theoretic essence of the technique introduced by Rotem and Segev [RS21] in the form of a stand-alone probabilistic lower bound. As in the stand-alone forking lemma of Bellare and Neven [BN06], our lemma considers a randomized algorithm $\mathcal{A}$ that is provided with $q+1$ inputs, where $q \geq 1$ may be any integer. Its first input is a value $x \in \mathcal{X}$, and its additional q inputs are values $h_1, \ldots, h_q \in \mathcal{C}$, for finite sets $\mathcal{X}$ and $\mathcal{C}$. Given such input, the algorithm then outputs a pair $(I, \sigma) \in \{0, \ldots, q\} \times \{0,1\}^*$. We are interested in the distribution of the output pair (I, σ), where the value x is sampled from a given distribution X over the set $\mathcal{X}$, and the values $h_1, \ldots, h_q$ are sampled independently and uniformly from the set $\mathcal{C}$. We let $X \times \mathcal{C}^q$ denote the corresponding product distribution.

For any such algorithm $\mathcal{A}$, and for any integer $B \geq 1$, we define the following "forking" algorithm $F_{\mathcal{A},B}$ that is given as input a value $x \in \mathcal{X}$ (note that the case $B = 1$ corresponds to the forking algorithm of Bellare and Neven [BN06]):

The Algorithm $\boldsymbol{F_{\mathcal{A},B}(x)}$

1. Sample $h_1, \ldots, h_q \leftarrow \mathcal{C}$ and $\rho \leftarrow \{0,1\}^*$ independently and uniformly.
2. Compute $(I_0, \sigma) = \mathcal{A}(x, h_1, \ldots, h_q; \rho)$.
3. If $I_0 \notin \{1, \ldots, q\}$ then output $\perp$ and terminate.
4. For any $j \in [B]$ sample $h_{I_0}^{(j)}, \ldots, h_q^{(j)} \leftarrow \mathcal{C}$ independently and uniformly, and compute
$$(I_j, \sigma_j) = \mathcal{A}\big(x, h_1, \ldots, h_{I_0 - 1}, h_{I_0}^{(j)}, \ldots, h_q^{(j)}; \rho\big).$$
5. If there exists an index $j \in [B]$ for which $I_j = I_0$ and $h_{I_j}^{(j)} \neq h_{I_0}$, then output
$$\big(I_0, \sigma, \sigma_j, h_1, \ldots, h_q, h_{I_0}^{(j)}, \ldots, h_q^{(j)}\big)$$
for the minimal such j, and otherwise output $\perp$.

Recall that, as defined in Section 2.1, for any $x \in \mathcal{X}$ we denote by $\mathsf{Time}(F_{\mathcal{A},B}(x))$ the random variable corresponding to the running time of the computation $F_{\mathcal{A},B}(x; h_1, \ldots, h_q, \rho)$ over the uniform choice of the randomness $(h_1, \ldots, h_q, \rho) \leftarrow \mathcal{C}^q \times \{0,1\}^*$. Equipped with our forking algorithm $F_{\mathcal{A},B}$, we prove the following lemma:

Lemma 1. *Let $q \geq 1$ and let $\mathcal{A}$ be a randomized algorithm that obtains $q+1$ inputs with associated finite sets $\mathcal{X}$ and $\mathcal{C}$ as described above. In addition, let t be an upper bound on the worst-case running time of $\mathcal{A}$, let X be a distribution over $\mathcal{X}$, and let*

$$\epsilon = \Pr_{(x, h_1, \ldots, h_q) \leftarrow X \times \mathcal{C}^q} \left[\mathcal{A}(x, h_1, \ldots, h_q) = (I, \sigma) \text{ s.t. } I \in \{1, \ldots, q\}\right].$$

If $\epsilon > 2 \cdot q^2/|\mathcal{C}|$ then for any $d \geq 1$ it holds that

$$\Pr_{x \leftarrow X}\left[F_{\mathcal{A},B}(x) \neq \perp\right] \geq \frac{B}{8q} \cdot \epsilon^2,$$

and

$$\mathbb{E}_{x \leftarrow X}\left[\left(\mathsf{Time}(F_{\mathcal{A},B}(x))\right)^d\right] \leq 2 \cdot (1+B)^d \cdot t^d \cdot \epsilon,$$

where $B = \left\lceil (1/\epsilon)^{1/d} - 1 \right\rceil$.

Proof of Lemma 1. We first prove the above lower bound on the success probability of $F_{\mathcal{A},B}(x)$ as a function of ϵ, B and q. For every $i \in \{1, \ldots, q\}$ and $h_1, \ldots, h_q \in \mathcal{C}$ we let $\vec{h}_i = (h_1, \ldots, h_i)$, and we let $\vec{h}_0 = \perp$. Based on the description of the algorithm $F_{\mathcal{A},B}$, it holds that $F_{\mathcal{A},B}(x) \neq \perp$ if and only if $I_0 \in \{1, \ldots, q\}$ and there exists an index $j \in [B]$ for which $I_j = I_0$ and $h_{I_j}^{(j)} \neq h_{I_0}$. Thus,

$$\begin{aligned}
&\Pr\left[F_{\mathcal{A},B}(x) \neq \perp\right] \\
&\quad = \Pr\left[(I_0 \in \{1, \ldots, q\}) \wedge \left(\bigvee_{j=1}^{B} (I_j = I_0) \wedge \left(h_{I_j}^{(j)} \neq h_{I_0}\right)\right)\right] \\
&\quad = \sum_{i=1}^{q} \Pr\left[(I_0 = i) \wedge \left(\bigvee_{j=1}^{B} \left(I_j = i \wedge h_i^{(j)} \neq h_i\right)\right)\right] \\
&\quad = \sum_{i=1}^{q} \sum_{\substack{(x,\rho) \in \mathcal{X} \times \{0,1\}^* \\ \vec{h}_{i-1} \in \mathcal{C}^{i-1}}} \left(\Pr\left[x \wedge \rho \wedge \vec{h}_{i-1}\right] \right.
\end{aligned}$$

$$\times \Pr\left[I_0 = i \wedge \left(\bigvee_{j=1}^{B}\Big(I_j = i \wedge h_i^{(j)} \neq h_i\Big)\right)\right]\Bigg)$$

$$\geq \sum_{i=1}^{q} \sum_{\substack{(x,\rho)\in\mathcal{X}\times\{0,1\}^* \\ \vec{h}_{i-1}\in\mathcal{C}^{i-1}}} \Bigg(\Pr\Big[x \wedge \rho \wedge h_{\vec{i-1}}\Big]$$

$$\times \Pr\left[I_0 = i \wedge \left(\bigvee_{j=1}^{B}\Big(I_j = i \wedge h_\ell^{(j)} \neq h_\ell \forall \ell \in \{i,\dots,q\}\Big)\right)\right]\Bigg).$$

For every $i \in \{1,\dots,q\}$, $x \in \mathcal{X}$, $\rho \in \{0,1\}^*$ and $\vec{h}_{i-1} \in \mathcal{C}^{i-1}$ we let $h_i^*(i,x,\rho,\vec{h}_{i-1}),\dots,$ $h_q^*(i,x,\rho,\vec{h}_{i-1})$ denote the lexicographically first $q-i+1$ elements of $\mathcal{C}$ such that the output of $\mathcal{A}$ on input $\big(x,\vec{h}_{i-1},h_i^*(i,x,\rho,\vec{h}_{i-1}),\dots,h_q^*(i,x,\rho,\vec{h}_{i-1});\rho\big)$ is a pair (I,σ) for which $I > 0$. At this point, since each value $h_\ell^{(j)}$ for $\ell \in \{i,\dots,q\}$ is uniformly sampled conditioned on x, ρ and $\vec{h}_{i-1}$, then instead of comparing it to the actual value h_ℓ we can compare it to the lexicographically first such h_ℓ^* that produces $I > 0$. Then,

$$\Pr\left[F_{\mathcal{A},B}(x) \neq \perp\right]$$

$$\geq \sum_{i=1}^{q} \sum_{\substack{(x,\rho)\in\mathcal{X}\times\{0,1\}^* \\ \vec{h}_{i-1}\in\mathcal{C}^{i-1}}} \Pr\Big[x \wedge \rho \wedge \vec{h}_{i-1}\Big]$$

$$\times \Pr\left[I_0 = i \wedge \left(\bigvee_{j=1}^{B}\Big(I_j = i \wedge h_\ell^{(j)} \neq h_\ell^*(i,x,\rho,\vec{h}_{i-1})\forall \ell \in \{i,\dots,q\}\Big)\right)\right]$$

$$= \sum_{i=1}^{q} \sum_{\substack{(x,\rho)\in\mathcal{X}\times\{0,1\}^* \\ \vec{h}_{i-1}\in\mathcal{C}^{i-1}}} \Pr\Big[x \wedge \rho \wedge \vec{h}_{i-1}\Big] \cdot \Pr\left[I_0 = i\right]$$

$$\times \Pr\left[\bigvee_{j=1}^{B}\Big(I_j = i \wedge h_\ell^{(j)} \neq h_\ell^*(i,x,\rho,\vec{h}_{i-1})\forall \ell \in \{i,\dots,q\}\Big)\right] \tag{1}$$

$$= \sum_{i=1}^{q} \sum_{\substack{(x,\rho)\in\mathcal{X}\times\{0,1\}^* \\ \vec{h}_{i-1}\in\mathcal{C}^{i-1}}} \Pr\Big[x \wedge \rho \wedge \vec{h}_{i-1}\Big] \cdot \Pr\left[I_0 = i\right])$$

$$\times \left(1 - \Pr\left[\bigwedge_{j=1}^{B}(I_j \neq i) \vee \Big(\exists \ell \in \{i,\dots,q\} \text{ s.t. } h_\ell^{(j)} = h_\ell^*(i,x,\rho,\vec{h}_{i-1}\Big)\right]\right)$$

$$= \sum_{i=1}^{q} \sum_{\substack{(x,\rho)\in\mathcal{X}\times\{0,1\}^* \\ \vec{h}_{i-1}\in\mathcal{C}^{i-1}}} \Pr\Big[x \wedge \rho \wedge \vec{h}_{i-1}\Big] \cdot \Pr\left[I_0 = i\right]$$

$$\times \left(1 - \prod_{j=1}^{B}\Pr\Big[I_j \neq i \vee \exists \ell \in \{i,\dots,q\} \text{ s.t. } h_\ell^{(j)} = h_\ell^*(i,x,\rho,\vec{h}_{i-1})\Big]\right) \tag{2}$$

where Eq. (1) follows from the fact that the events $I_0 = i$ and

$$\bigvee_{j=1}^{B} (I_j = i) \wedge h_\ell^{(j)} \neq h_\ell^*(i, x, \rho, \vec{h}_{i-1}) \forall \ell \in \{i, .., q\}$$

are independent conditioned on x, ρ and $\vec{h}_{i-1}$, and Eq. (2) follows from the fact that for any $1 \leq j_1 \neq j_2 \leq B$ the corresponding executions of $\mathcal{A}$ are independent. Now, via a union bound we obtain

$$\begin{aligned}\Pr&\left[I_j \neq i \vee \exists \ell \in \{i, \ldots, q\} \text{ s.t. } h_\ell^{(j)} = h_\ell^*(i, x, \rho, \vec{h}_{i-1})\right] \\ &\leq \min\left\{1, \Pr\left[I_j \neq i\right] + \Pr\left[\exists \ell \in \{i, \ldots, q\} \text{ s.t. } h_\ell^{(j)} = h_\ell^*(i, x, \rho, \vec{h}_{i-1})\right]\right\} \\ &\leq \min\left\{1, 1 - \Pr\left[I_j = i\right] + \frac{q}{|\mathcal{C}|}\right\}.\end{aligned}$$

For every i, x, ρ, and $h_1, \ldots, h_{i-1}$ we let

$$\tilde{\epsilon_i}(x, \rho, \vec{h}_{i-1}) = \max\left\{0, \Pr\left[I_0 = i\right] - \frac{q}{|\mathcal{C}|}\right\},$$

then $\Pr\left[I_0 = i\right] \geq \tilde{\epsilon_i}(x, \rho, \vec{h}_{i-1})$ and $\Pr\left[I_0 = i\right] = \Pr\left[I_j = i\right]$ for every $j \in [B]$. Thus,

$$\begin{aligned}1 - \prod_{j=1}^{B} \Pr&\left[I_j \neq i \vee \exists \ell \in \{i, \ldots, q\} \text{ s.t. } h_\ell^{(j)} = h_\ell^*(i, x, \rho, \vec{h}_{i-1})\right] \\ &\geq 1 - \left(1 - \tilde{\epsilon_i}(x, \rho, \vec{h}_{i-1})\right)^B\end{aligned}$$

and therefore

$$\begin{aligned}\Pr\left[F_{\mathcal{A},B}(x) \neq \bot\right] &\geq \sum_{i=1}^{q} \sum_{\substack{(x,\rho) \in \mathcal{X} \times \{0,1\}^* \\ \vec{h}_{i-1} \in \mathcal{C}^{i-1}}} \Pr\left[x \wedge \rho \wedge \vec{h_{i-1}}\right] \\ &\qquad \times \tilde{\epsilon_i}(x, \rho, \vec{h}_{i-1}) \cdot \left(1 - \left(1 - \tilde{\epsilon_i}(x, \rho, \vec{h}_{i-1})\right)^B\right) \\ &= \sum_{i=1}^{q} \mathbb{E}\left[\tilde{\epsilon_i}(x, \rho, \vec{h}_{i-1}) \cdot \left(1 - \left(1 - \tilde{\epsilon_i}(x, \rho, \vec{h}_{i-1})\right)^B\right)\right] \\ &\geq \frac{1}{2} \cdot B \cdot \sum_{i=1}^{q} \tilde{\epsilon_i}^2, \qquad (3)\end{aligned}$$

where we let $\tilde{\epsilon_i} = \mathbb{E}\left[\tilde{\epsilon_i}(x, \rho, \vec{h}_{i-1})\right]$ for every $i \in [q]$, and Eq. (3) follows from the following claim which is proved in Appendix A.1:

Claim 3. *For each $i \in [q]$ it holds that*

$$\mathbb{E}\left[\tilde{\epsilon_i}(x, \rho, \vec{h}_{i-1}) \cdot \left(1 - \left(1 - \tilde{\epsilon_i}(x, \rho, \vec{h}_{i-1})\right)^B\right)\right] \geq \frac{1}{2} \cdot B \cdot \tilde{\epsilon_i}^2$$

Letting $\epsilon_i = \Pr[I_0 = i]$ for every $i \in [q]$, and using Jensen's inequality, we obtain:

$$\begin{aligned}
\Pr[F_{\mathcal{A},B}(x) \neq \bot] &\geq \frac{1}{2} \cdot B \cdot \sum_{i=1}^{q} \tilde{\epsilon_i}^2 \\
&\geq \frac{1}{2q} \cdot B \cdot \left(\sum_{i=1}^{q} \tilde{\epsilon_i} \right)^2 \\
&\geq \frac{1}{2q} \cdot B \cdot \left(\sum_{i=1}^{q} \left(\epsilon_i - \frac{q}{|\mathcal{C}|} \right) \right)^2 \qquad (4) \\
&= \frac{1}{2q} \cdot B \cdot \left(\Pr[I_0 \in \{1, \ldots, q\}] - \frac{q^2}{|\mathcal{C}|} \right)^2
\end{aligned}$$

where Eq. (4) follows from the following claim which is proved in Appendix A.2:

Claim 4. *For every $i \in [q]$ it holds that $\tilde{\epsilon_i} \geq \epsilon_i - \frac{q}{|\mathcal{C}|}$.*

Now, we obtain

$$\begin{aligned}
\Pr[F_{\mathcal{A},B}(x) \neq \bot] &\geq \frac{1}{2q} \cdot B \cdot \left(\Pr[I_0 \in \{1, \ldots, q\}] - \frac{q^2}{|\mathcal{C}|} \right)^2 \\
&\geq \frac{1}{2q} \cdot B \cdot \left(\epsilon - \frac{q^2}{|\mathcal{C}|} \right)^2 \\
&\geq \frac{1}{2q} \cdot B \cdot \left(\frac{\epsilon}{2} \right)^2 \qquad (5) \\
&= \frac{1}{8q} \cdot B \cdot \epsilon^2,
\end{aligned}$$

as required, where Eq. (5) follows from the assumption $\epsilon > 2 \cdot q^2/|\mathcal{C}|$.

We now turn to upper bounding the d-th moment of the running time of the algorithm $F_{\mathcal{A},B}$. Note that, when $x \leftarrow X$, then with probability $1 - \epsilon$ the algorithm $F_{\mathcal{A},B}$ runs in time at most t, and with probability ϵ it runs in time at most $(1 + B) \cdot t$. Therefore,

$$\begin{aligned}
\mathbb{E}_{x \leftarrow X}\left[(\mathsf{Time}(F_{\mathcal{A},B}(x)))^d \right] &\leq (1 - \epsilon) \cdot t^d + \epsilon \cdot ((1 + B) \cdot t)^d \\
&\leq t^d \cdot \epsilon \cdot (1 + B)^d + \epsilon \cdot (1 + B)^d \cdot t^d \\
&= 2 \cdot (1 + B)^d \cdot t^d \cdot \epsilon, \qquad (6)
\end{aligned}$$

where Eq. (6) following from our choice of $B \geq (1/\epsilon)^{\frac{1}{d}} - 1$, which implies that $1 \leq \epsilon \cdot (1 + B)^d$. This concludes the proof of Lemma 1. ■

4 Tighter Concrete Security for BN Multi-Signatures

In this section we show that our high-moment forking lemma can be used for establishing a concrete security bound for the Bellare-Neven (BN) multi-signature scheme [BN06]. Our starting point is the recent work of Bellare and Dai [BD21], who showed that the security of the Bellare-Neven multi-signature scheme is equivalent to the hardness of the identification discrete-logarithm (IDL) problem, introduced by Kiltz, Masny, and Pan [KMP16]. Bellare and Dai showed that in the algebraic-group model [FKL18] the hardness of the IDL problem is equivalent to that of the DL problem. As previously discussed, the algebraic group model, however, is an idealized model which considers a rather restricted class of attackers (attackers which are assumed to provide an algebraic justification of each group element that they produce). In fact, in this model, the hardness of an extremely wide class of strong computational and decisional problems is known to be equivalent to that of the DL problem [FKL18, AHK20, BFL20, FPS20, MTT19, RS20]).

In the standard model, Kiltz, Masny, and Pan [KMP16] showed that the hardness of the IDL problem can be based on that of the DL problem using the forking lemma, once again leading to the square-root loss for the BN multi-signature scheme. By applying our high-moment forking lemma, assuming the second-moment hardness of the DL problem, we prove a tighter concrete hardness bound for the IDL problem in the standard model and, therefore, also for the Bellare-Neven multi-signature scheme.

For stating our theorem, recall that the BN scheme relies on a cyclic group $\mathbb{G}$ of prime order p that is generated by a given element $g \in \mathbb{G}$, as well as on an integer ℓ. We let $\mathcal{G} = (\mathbb{G}, p, g)$ denote the description of the group, and we let $\Pi_{\mathsf{BN}[\mathcal{G},\ell]}$ denote the BN scheme. The scheme relies on two hash functions, H_0 and H_1, that are modeled as random oracles for the scheme's security analysis. These two hash functions are used to map arbitrary strings to the sets $\{0,1\}^\ell$ and $\mathbb{Z}_p$. That is, $H_0 : \{0,1\}^* \to \{0,1\}^\ell$ and $H_1 : \{0,1\}^* \to \mathbb{Z}_p$. We prove the following theorem:

Theorem 5. *Let $\mathcal{G} = (\mathbb{G}, p, g)$, where $\mathbb{G}$ be a cyclic group of prime order p that is generated by an element $g \in \mathbb{G}$, and let ℓ be an integer. Assuming that the DL problem is second-moment hard in the group $\mathbb{G}$, then for any adversary A it holds that*

$$\mathsf{Adv}^{\mathsf{MultiSig}}_{\Pi_{\mathsf{BN}[\mathcal{G},\ell]}}(A) \leq \left(32 \cdot \frac{q_1 \cdot t^2}{p}\right)^{2/3} + \frac{q_{\mathsf{S}} \cdot (4q_0 + 2q_1 + q_{\mathsf{S}})}{p} + \frac{q_0 \cdot (q_0 + n)}{2^\ell},$$

where:

- *t is an upper bound on the running time of the experiment $\mathsf{Exp}^{\mathsf{MultiSig}}_{\Pi_{\mathsf{BN}[\mathcal{G},\ell]}}(A)$.*
- *q_0, q_1 and q_{S} are upper bounds on the number of H_0-queries, H_1-queries and signing queries issued during the experiment $\mathsf{Exp}^{\mathsf{MultiSig}}_{\Pi_{\mathsf{BN}[\mathcal{G},\ell]}}(A)$, respectively.*
- *n is an upper bound on the largest signer set involved in a single multi-signature during the experiment $\mathsf{Exp}^{\mathsf{MultiSig}}_{\Pi_{\mathsf{BN}[\mathcal{G},\ell]}}(A)$.*

Compared to the bound proved by Bellare and Dai [BD21], our bound is tighter by replacing their exponent 1/2 with our exponent 2/3. All other terms in our bound are exactly the same as in their bound (up to the multiplicative constant 32), as we explain in Section 4.2. In addition, note that as with the bound of Bellare and Dai, our bound is stated in terms of the parameters t, q_0, q_1 and q_{S} of the entire experiment $\mathsf{Exp}^{\mathsf{MultiSig}}_{\Pi_{\mathsf{BN}[\mathcal{G},\ell]}}(A)$ and not those of the adversary only (e.g., the number of H_0 and H_1 queries issued during the experiment consists of the number of such queries issued directly by the adversary and the number of such queries issued by the signing oracle).

In what follows we first describe the BN scheme $\Pi_{\mathsf{BN}[\mathcal{G},\ell]}$. Then, in Section 4.1 we prove a tighter concrete hardness bound for the IDL problem in the standard model, which we use in Section 4.2 for deriving Theorem 5. For simplicity, in the following description of the scheme $\Pi_{\mathsf{BN}[\mathcal{G},\ell]}$, we assume that all algorithms receive $\mathcal{G} = (\mathbb{G}, p, g)$ and $\ell \in \mathbb{N}$ as inputs, and we explicitly include them only for the key-generation algorithm.

The BN Multi-Signature Scheme $\Pi_{\mathsf{BN}[\mathcal{G},\ell]}$

$\mathsf{KG}(\mathcal{G},\ell)$**.** On input $\mathcal{G} = (\mathbb{G}, p, g)$ and $\ell \in \mathbb{N}$, the key-generation algorithm samples $x \leftarrow \mathbb{Z}_p$, and outputs

$$(\mathsf{sk}, \mathsf{vk}) = (x, g^x) \in \mathbb{Z}_p \times \mathbb{G}.$$

$\mathsf{Sign}\left(\mathsf{sk}, \mathsf{vk}, \vec{\mathsf{vk}}, m\right)$**.** On input sk and vk as above, $\vec{\mathsf{vk}} = (\mathsf{vk}_1, \ldots, \mathsf{vk}_n)$ for some $n \in \mathbb{N}$, and $m \in \{0,1\}^*$, the signing process is defined as the following 3-round protocol:

1. If there is no index $i \in [n]$ for which $\mathsf{vk} = \mathsf{vk}_i$ or if there is more than one such index, then abort. Otherwise, denote by $i^* \in [n]$ the unique such index.
2. Sample $r_{i^*} \leftarrow \mathbb{Z}_p$, let $R_{i^*} = g^{r_{i^*}}$, and send $t_{i^*} = H_0(i^*, R_{i^*}) \in \{0,1\}^\ell$ to all other parties.
3. Upon receiving $\{t_i\}_{i\in[n]\setminus\{i^*\}}$ from all other parties, send R_{i^*} to all other parties.
4. Upon receiving $\{R_i\}_{i\in[n]\setminus\{i^*\}}$ from all other parties, if for some $i \in [n] \setminus \{i^*\}$ it holds that $t_i \neq H_0(i, t_i)$ then abort. Otherwise, compute

$$\begin{aligned} R &= \prod_{i\in[n]} R_i \in \mathbb{G} \\ c_{i^*} &= H_1\left(\mathsf{vk}, \vec{\mathsf{vk}}, R, m\right) \in \mathbb{Z}_p \\ s_{i^*} &= r_{i^*} + \mathsf{sk} \cdot c_{i^*} \bmod p, \end{aligned}$$

and locally output (R, s_{i^*}).

$\mathsf{SigAgg}\left((R_1, s_1), \ldots, (R_n, s_n)\right)$**.** On input $(R_1, s_1), \ldots, (R_n, s_n) \in \mathbb{G} \times \mathbb{Z}_p$ for some $n \in \mathbb{N}$, if $R_1 = \cdots = R_n$ then compute $s = \sum_{i\in[n]} s_i \bmod p$ and output $\sigma = (R, s)$. Otherwise output $\perp$.

$\mathsf{Verify}\left(\vec{\mathsf{vk}}, m, \sigma\right)$**.** On input $\vec{\mathsf{vk}} = (\mathsf{vk}_1, \ldots, \mathsf{vk}_n)$ for some $n \in \mathbb{N}$, $m \in \{0,1\}^*$ and $\sigma = (R, s) \in \mathbb{G} \times \mathbb{Z}_p$, for every $i \in [n]$ compute

$$c_i = H_1\left(\mathsf{vk}_i, \vec{\mathsf{vk}}, R, m\right) \in \mathbb{Z}_p,$$

and output 1 if and only if $g^s = R \cdot \prod_{i\in[n]} (\mathsf{vk}_i)^{c_i}$.

4.1 IDL Hardness Based on Second-Moment DL Hardness

The identification discrete logarithm (IDL) problem, introduced by Kiltz, Masny, and Pan [KMP16] and further studied by Bellare and Dai [BD21], is parameterized by an integer q and a description $(\mathbb{G}, p, g)$ of a cyclic group as above. It considers an algorithm A that is provided with a uniformly distributed group element $X \in \mathbb{G}$ as input and may issue up to q queries to the following oracle $\mathcal{O}_{\mathsf{IDL}}$: On input as query a group element $R \in \mathbb{G}$, it samples and returns a challenge c, which is distributed uniformly and independently of all previous queries. Denoting by $R_1, \ldots, R_q$ the queries issued by A, and by $c_1, \ldots, c_q$ the corresponding challenges returned by the oracle, the goal of the algorithm A is to output a pair (I, z) for which $I \in [q]$ and $g^z = R_I \cdot X^{c_I}$. The advantage of such an algorithm A is formally captured by the following definition:

Definition 4. Let $\mathcal{G} = (\mathbb{G}, p, g)$, where $\mathbb{G}$ be a cyclic group of prime order p that is generated by an element $g \in \mathbb{G}$, and let q be an integer. For any algorithm A that issues at most q oracle queries, we define

$$\mathsf{Adv}^{\mathsf{IDL}}_{\mathcal{G},q}(A) \overset{\mathsf{def}}{=} \Pr\left[\mathsf{Exp}^{\mathsf{IDL}}_{\mathcal{G},q}(A) = 1\right],$$

where the experiment $\mathsf{Exp}^{\mathsf{IDL}}_{\mathcal{G},q}(A)$ is defined as follows:

1. $X \leftarrow \mathbb{G}$.

2. $(I, z) \leftarrow A^{\mathcal{O}_{\mathsf{IDL}}(\cdot)}(X)$.
3. Denote by $R_1, \ldots, R_q \in \mathbb{G}$ the oracle queries issued by A, and by $c_1, \ldots, c_q \in \mathbb{Z}_p$ the corresponding responses.
4. If $I \in [q]$ and $g^z = R_I \cdot X^{c_I}$ then output 1 and otherwise output 0.

The following lemma shows that our high-moment forking lemma can be applied to derive a concrete hardness result for the IDL problem which improved upon the above-discussed square-root loss without relying on idealized models [KMP16, BD21].

Lemma 2. *Let $\mathcal{G} = (\mathbb{G}, p, g)$, where $\mathbb{G}$ be a cyclic group of prime order p that is generated by an element $g \in \mathbb{G}$, let A_{IDL} be an algorithm that runs in time at most t and issues at most q oracle queries, and let*

$$\epsilon = \mathsf{Adv}^{\mathsf{IDL}}_{\mathcal{G},q}(A_{\mathsf{IDL}}).$$

Then, either $\epsilon < \frac{2 \cdot q^2}{p}$, or there exists an algorithm A_{DL} such that

$$\Pr_{x \leftarrow \mathbb{Z}_p}\left[A_{\mathsf{DL}}\left(\mathcal{G}, g^x\right) = x\right] \geq \frac{B}{8q} \cdot \epsilon^2$$

and

$$\mathbb{E}_{x \leftarrow \mathbb{Z}_p}\left[\left(\mathsf{Time}(A_{\mathsf{DL}}\left(\mathcal{G}, g^x\right))\right)^2\right] \leq 2 \cdot (1 + B)^2 \cdot t^2 \cdot \epsilon,$$

for $B = \left\lceil (1/\epsilon)^{1/2} - 1 \right\rceil$.

Equipped with Lemma 2, we directly obtain the following corollary based on the second-moment hardness of the DL problem:

Corollary 1. *Let $\mathcal{G} = (\mathbb{G}, p, g)$, where $\mathbb{G}$ be a cyclic group of prime order p that is generated by an element $g \in \mathbb{G}$, and let A be an algorithm that runs in time at most t and issues at most q oracle queries. Then, assuming that the DL problem is second-moment hard in the group $\mathbb{G}$, it holds that*

$$\mathsf{Adv}^{\mathsf{IDL}}_{\mathcal{G},q}(A) \leq \left(32 \cdot \frac{q \cdot t^2}{p}\right)^{2/3}.$$

In what follows we prove Lemma 2 and Corollary 1.

Proof of Lemma 2. Let $\mathcal{G} = (\mathbb{G}, p, g)$, where $\mathbb{G}$ be a cyclic group of prime order p that is generated by an element $g \in \mathbb{G}$, and let A_{IDL} be an algorithm that runs in time at most t and issues at most q oracle queries. First, we transform the algorithm A_{IDL} into an algorithm A'_{IDL} which is compatible with the Lemma 1. The algorithm A'_{IDL} receives as input the group description $\mathcal{G} = (\mathbb{G}, p, g)$, a group element $X \in \mathbb{G}$ and values $c_1, \ldots, c_q \in \mathbb{Z}_p$, as well as randomness $r \in \{0,1\}^*$ of the appropriate length for running A_{IDL}, and is defined as follows:

The Algorithm $A'_{\mathsf{IDL}}(\mathcal{G}, X, c_1, \ldots., c_q; r)$

1. Invoke $A_{\mathsf{IDL}}(\mathcal{G}, X; r)$.
2. For every $i \in [q]$, when A_{IDL} issues its ith oracle query R_i, respond with c_i.
3. If A_{IDL} outputs (I, z) such that $I \in [q]$ and $g^z = R_I \cdot X^{c_I}$ then output $(I, (z, c_I))$, and otherwise output $(0, \perp)$.

That is, the algorithm A'_{IDL} emulates the experiment $\mathsf{Exp}^{\mathsf{IDL}}_{\mathcal{G},q}(A_{\mathsf{IDL}})$ while using the values $c_1, \ldots, c_q$ as the responses of the oracle. Therefore, the running time of A'_{IDL} is identical to the running time t of A_{IDL} (for simplicity we ignore an additional minor additive term due to the verification of the equation $g^z = R_I \cdot X^{c_I}$), and it holds that

$$\Pr_{(X,c_1,...,c_q)\leftarrow \mathbb{G}\times(\mathbb{Z}_p)^q}\left[A'_{\mathsf{IDL}}(\mathcal{G},X,c_1,...,c_q) = (I,(z,c_I)) \text{ s.t. } I>0\right] = \mathsf{Adv}^{\mathsf{IDL}}_{\mathcal{G},q}(A_{\mathsf{IDL}}) = \epsilon.$$

Now, assuming that $\epsilon \geq \frac{2\cdot q^2}{p}$, we apply Lemma 1 with the algorithm A'_{IDL}, the parameters q and $d = 2$, and with the sets $\mathcal{X} = \mathbb{G}$ and $\mathcal{C} = \mathbb{Z}_p$. This yields an algorithm F such that

$$\Pr_{X\leftarrow\mathbb{G}}\left[F(\mathcal{G},X) \neq \perp\right] \geq \frac{B}{8q}\cdot\epsilon^2,$$

and

$$\mathbb{E}_{X\leftarrow\mathbb{G}}\left[\left(\mathsf{Time}_{F(\mathcal{G},X)}\right)^2\right] \leq 2\cdot(1+B)^2\cdot t^2\cdot\epsilon,$$

where $B = \left\lceil (1/\epsilon)^{1/2} - 1\right\rceil$. Equipped with the algorithm F, we can now define the following DL algorithm A_{DL}:

The Algorithm $A_{\mathsf{DL}}(\mathcal{G}, X)$

1. Invoke $F(\mathcal{G},X)$ and output $\perp$ if F returned $\perp$.
2. Otherwise, denote by $(I,(z,c_I),(z',c'_I))$ the output of F, and output

$$x = (c_I - c'_I)^{-1}\cdot(z-z') \bmod p.$$

Note that the distribution of A_{DL}'s running time is identical to that of F (for simplicity we are ignoring an additional minor additive term due to the computation of x). In addition, note that whenever F returns $(I,(z,c_I),(z',c'_I))$ then $c_I \neq c'_I$ (thus $c_I - c'_I$ can indeed be inverted modulo p), and it holds that $g^z = R_I\cdot X^{c_I}$ and $g^{z'} = R_I\cdot X^{c'_I}$ for some $R_I\in\mathbb{G}$. Therefore, it holds that $X = g^x$ for $x = (c_I - c'_I)^{-1}\cdot(z-z') \bmod p$, and

$$\begin{aligned}\Pr_{x\leftarrow\mathbb{Z}_p}\left[A_{\mathsf{DL}}\left(\mathcal{G},g^x\right) = x\right] &\geq \Pr_{X\leftarrow\mathbb{G}}\left[F(\mathcal{G},X)\neq\perp\right]\\ &\geq \frac{B}{8q}\cdot\epsilon^2.\end{aligned}$$

■

Proof of Corollary 1. Let $\mathcal{G} = (\mathbb{G},p,g)$, where $\mathbb{G}$ be a cyclic group of prime order p that is generated by an element $g\in\mathbb{G}$, let A_{IDL} be an algorithm that runs in time at most t and issues at most q oracle queries, and let

$$\epsilon = \mathsf{Adv}^{\mathsf{IDL}}_{\mathcal{G},q}(A_{\mathsf{IDL}}).$$

Lemma 2 stated that either $\epsilon < \frac{2\cdot q^2}{p}$, or that there exists an algorithm A_{DL} such that

$$\Pr_{x\leftarrow\mathbb{Z}_p}\left[A_{\mathsf{DL}}\left(\mathcal{G},g^x\right) = x\right] \geq \frac{B}{8q}\cdot\epsilon^2$$

and

$$\mathbb{E}_{x\leftarrow\mathbb{Z}_p}\left[\left(\mathsf{Time}(A_{\mathsf{DL}}\left(\mathcal{G},g^x\right))\right)^2\right] \leq 2\cdot(1+B)^2\cdot t^2\cdot\epsilon,$$

for $B = \left\lceil (1/\epsilon)^{1/2} - 1\right\rceil$. Assuming that the DL problem is second moment hard in the group $\mathbb{G}$ guarantees that

$$\Pr_{x\leftarrow\mathbb{Z}_p}\left[A_{\mathsf{DL}}\left(\mathcal{G},g^x\right) = x\right] \leq \frac{\mathbb{E}_{x\leftarrow\mathbb{Z}_p}\left[\left(\mathsf{Time}(A_{\mathsf{DL}}\left(\mathcal{G},g^x\right))\right)^2\right]}{p},$$

and thus

$$\frac{B}{8q} \cdot \epsilon^2 \leq \frac{2 \cdot (1+B)^2 \cdot t^2 \cdot \epsilon}{p}.$$

This implies that

$$\epsilon \leq \left(32 \cdot \frac{q \cdot t^2}{p}\right)^{2/3}.$$

Taking into account also the case in which $\epsilon < \frac{2 \cdot q^2}{p}$, and the fact that $q \leq t$, we obtain that

$$\epsilon \leq \max \left\{ \left(32 \cdot \frac{q \cdot t^2}{p}\right)^{2/3}, \frac{2 \cdot q^2}{p} \right\} = \left(32 \cdot \frac{q \cdot t^2}{p}\right)^{2/3},$$

which settles the proof of Corollary 1. ■

4.2 Proof of Theorem 5

For deriving Theorem 5 we rely on the following theorem due to Bellare and Dai [BD21] (refining the original analysis of Bellare and Neven [BN06]):

Theorem 6 ([BD21])**.** *Let $\mathcal{G} = (\mathbb{G}, p, g)$, where $\mathbb{G}$ be a cyclic group of prime order p that is generated by an element $g \in \mathbb{G}$, and let ℓ be an integer. For any adversary A that exists an adversary A' that such*

$$\mathsf{Adv}^{\mathsf{MultiSig}}_{\Pi_{\mathsf{BN}[\mathcal{G},\ell]}}(A) \leq \mathsf{Adv}^{\mathsf{IDL}}_{\mathcal{G},q_1}(A') + \frac{q_{\mathsf{S}} \cdot (4q_0 + 2q_1 + q_{\mathsf{S}})}{p} + \frac{q_0 \cdot (q_0 + n)}{2^\ell},$$

for all sufficiently large $\lambda \in \mathbb{N}$, where:

- *The running time of A' is that of the experiment $\mathsf{Exp}^{\mathsf{MultiSig}}_{\Pi_{\mathsf{BN}[\mathcal{G},\ell]}}(A)$.*
- *q_0, q_1 and q_{S} are upper bounds on the number of H_0-queries, H_1-queries and signing queries issued during the experiment $\mathsf{Exp}^{\mathsf{MultiSig}}_{\Pi_{\mathsf{BN}[\mathcal{G},\ell]}}(A)$, respectively.*
- *n is an upper bound on the largest signer set involved in a single multi-signature during the experiment $\mathsf{Exp}^{\mathsf{MultiSig}}_{\Pi_{\mathsf{BN}[\mathcal{G},\ell]}}(A)$.*

Theorem 5 now easily follows by replacing the term $\mathsf{Adv}^{\mathsf{IDL}}_{\mathcal{G},q_1}(A')$ in the statement of Theorem 6 with the term $\left(32 \cdot \frac{q_1 \cdot t^2}{p}\right)^{2/3}$ provided by Corollary 1. ■

5 Tighter Concrete Security for BLS Multi-Signatures

In this section we show that our high-moment forking lemma can be used for establishing a concrete security bound for the BLS multi-signature scheme, introduced and analyzed by Boneh and Drijvers and Neven [BDN18]. For stating our theorem, recall that the BLS multi-signature scheme (which is described below) relies on an efficiently-computable non-degenerated bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$, where $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_t$ are cyclic groups of prime order p. We let $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, p, g_1, g_2, e)$ denote the description of the groups and of the bilinear map, where g_1 and g_2 are generators of the groups $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively, and we let $\Pi_{\mathsf{BLS}[\mathcal{G}]}$ denote the BLS multi-signature scheme. Additionally, the scheme relies on two hash functions, H_0 and H_1, that are modeled as random oracles for the scheme's security analysis. These two hash functions are used to map arbitrary strings to $\mathbb{G}_1$ and $\mathbb{Z}_p$. That is, $H_0 : \{0,1\}^* \to \mathbb{G}_1$ and $H_1 : \{0,1\}^* \to \mathbb{Z}_p$. We prove the following theorem:

Theorem 7. *Let $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, p, g_1, g_2, e)$ as above. Assuming that the co-CDH problem is second-moment hard in $\mathcal{G}$, then for any adversary A it holds that*

$$\mathsf{Adv}^{\mathsf{rMultiSig}}_{\Pi_{\mathsf{BLS}[\mathcal{G}]}}(A) \leq \left(16 q_0^{3/2} \cdot q_1 \cdot \frac{\left(t + q_0 \cdot \tau_{\mathsf{exp}_1} + q_{\mathsf{S}} \cdot \tau_{\mathsf{exp}_1} + \tau_{\mathsf{exp}_2^n}\right)^2}{p} \right)^{2/3},$$

where:

- *t is an upper bound on the running time of the adversary A.*
- *q_0, q_1 and q_{S} are upper bounds on the number of H_0-queries, H_1-queries and signing queries issued by the adversary A, respectively.*
- *n is an upper bound on the largest signer set involved in a single multi-signature during the experiment $\mathsf{Exp}^{\mathsf{rMultiSig}}_{\Pi_{\mathsf{BLS}[\mathcal{G}]}}(A)$.*
- *τ_{exp_1} is the time required to compute an exponentiation in $\mathbb{G}_1$, and $\tau_{\mathsf{exp}_2^n}$ is the time required to compute an n-multi-exponentiation in $\mathbb{G}_2$.*

Note that in the statement of the above theorem, unlike in the statement of Theorem 5 for the BN scheme, we have explicitly included the exponentiation times in $\mathbb{G}_1$ and $\mathbb{G}_2$. This is due to the fact that the exponentiation times in these two groups may be rather different, whereas for the BN scheme there is only one group.

Additionally, as discussed in Section 2, to enable a direct-as-possible comparison to the concrete security bound proved by Boneh and Drijvers and Neven [BDN18], we analyze the concrete security of the BLS scheme with respect to the relaxed security experiment $\mathsf{Exp}^{\mathsf{rMultiSig}}_{\Pi}$. This security experiment is obtained from the standard security experiment $\mathsf{Exp}^{\mathsf{MultiSig}}_{\Pi}$ for multi-signature schemes (see Definition 3) by asking that an adversary outputting $\left(\vec{\mathsf{vk}}^*, m^*, \mathsf{agg}\sigma^*\right)$ does not issue a signing query $\left(\vec{\mathsf{vk}}, m^*\right)$ for *any* vector $\vec{\mathsf{vk}}$ of verification keys. As discussed in Section 2, the BLS scheme can be easily transformed into one that satisfies the standard notion.

For comparing our bound to the one proved by Boneh and Drijvers and Neven (see [BDN18, Thm. 1]), note that Boneh and Drijvers and Neven transformed any malicious forger that runs in time t, issues $q_{\mathsf{H}} = q_0 + q_1$ random oracle queries and q_{S} signing queries, and breaks the security of the scheme with probability ϵ into a co-CDH algorithm that runs in time

$$\left(t + q_{\mathsf{H}} \cdot \tau_{\mathsf{exp}_1} + q_{\mathsf{S}} \cdot (\tau_{\mathsf{exp}_1} + \tau_{\mathsf{exp}_2^n})\right) \cdot \frac{8q_{\mathsf{H}}^2}{\epsilon} \cdot \ln\left(\frac{8q_{\mathsf{H}}}{\epsilon}\right)$$

and has success probability $\epsilon/(8q_{\mathsf{H}})$. Thus, in bilinear groups of prime order p in which one assumes that the co-CDH problem is as hard as in the generic bilinear-group model [Sho97, Mau05, BB08] (i.e., that any algorithm running in time T solves the co-CDH problem with probability at most T^2/p), we obtain (when ignoring for simplicity the lower-order logarithmic term as well as multiplicative constants) the bound

$$\epsilon \leq q_{\mathsf{H}}^{5/3} \cdot \left(\frac{\left(t + q_{\mathsf{H}} \cdot \tau_{\mathsf{exp}_1} + q_{\mathsf{S}} \cdot (\tau_{\mathsf{exp}_1} + \tau_{\mathsf{exp}_2^n})\right)^2}{p} \right)^{1/3}$$

on the success probability of any such malicious forger. In contrast, Theorem 7 (when replacing q_0 and q_1 with q_{H} that upper bounds them, and ignoring the multiplicative constant $2^{2/3}$), provides the bound

$$\epsilon \leq q_{\mathsf{H}}^{5/3} \cdot \left(\frac{\left(t + q_{\mathsf{H}} \cdot \tau_{\mathsf{exp}_1} + q_{\mathsf{S}} \cdot \tau_{\mathsf{exp}_1} + \tau_{\mathsf{exp}_2^n}\right)^2}{p} \right)^{2/3}$$

Thus, our bound is tighter by replacing their exponent 1/2 with our exponent 2/3 (with additional, although rather minor, improved dependence on the number of signing queries q_{S} and multi-exponentiation time $\tau_{\mathsf{exp}_2^n}$ in $\mathbb{G}_2$).

In what follows we first describe the $\Pi_{\mathsf{BLS}[\mathcal{G}]}$ scheme. Then, we show that our forking lemma enables to transform any forger that attacks the scheme into a co-CDH algorithm designed in order to optimize the trade-off between second-moment of their running time and their success probability. Finally, we show that Theorem 7 then follows by combining the resulting trade-off with the assumption that the co-CDH problem is second-moment hard. For simplicity, in the following description of the scheme $\Pi_{\mathsf{BLS}[\mathcal{G}]}$, we assume that all algorithms receive $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, p, g_1, g_2, e)$ as input, and we explicitly include it only for the key-generation algorithm.

The BLS Multi-Signature Scheme $\boldsymbol{\Pi_{\mathsf{BLS}[\mathcal{G}]}}$

$\mathsf{KG}(\mathcal{G})$. On input $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, p, g_1, g_2, e)$, the key-generation algorithm samples $x \leftarrow \mathbb{Z}_p$, and outputs

$$(\mathsf{sk}, \mathsf{vk}) = (x, g_2^x) \in \mathbb{Z}_p \times \mathbb{G}_2.$$

$\mathsf{KeyAgg}\left(\vec{\mathsf{vk}}\right)$. On input $\vec{\mathsf{vk}} = (\mathsf{vk}_1, \dots, \mathsf{vk}_n) \in \mathbb{G}_2^n$ for some $n \in \mathbb{N}$, for every $i \in [n]$ compute $t_i = H_1\left(\mathsf{vk}_i, \vec{\mathsf{vk}}\right) \in \mathbb{Z}_p$, and output

$$\mathsf{aggvk} = \prod_{i \in [n]} \mathsf{vk}_i^{t_i} \in \mathbb{G}_2.$$

$\mathsf{Sign}\left(\mathsf{sk}, \mathsf{vk}, \vec{\mathsf{vk}}, m\right)$. On input sk and vk as above, $\vec{\mathsf{vk}} = (\mathsf{vk}_1, \dots, \mathsf{vk}_n)$ for some $n \in \mathbb{N}$, and $m \in \{0,1\}^*$, compute $t = H_1\left(\mathsf{vk}, \vec{\mathsf{vk}}\right) \in \mathbb{Z}_p$ and output

$$s = H_0\left(m\right)^{t \cdot \mathsf{sk}} \in \mathbb{G}_1.$$

$\mathsf{SigAgg}\left(s_1, \dots, s_n\right)$. On input $s_1, \dots, s_n \in \mathbb{G}_1$ for some $n \in \mathbb{N}$, output

$$\sigma = \prod_{i \in [n]} s_i \in \mathbb{G}_1.$$

$\mathsf{Verify}\left(\mathsf{aggvk}, m, \sigma\right)$. On input $\mathsf{aggvk} \in \mathbb{G}_2$, $m \in \{0,1\}^*$ and $\sigma \in \mathbb{G}_1$, if

$$e\left(\sigma, g_2\right) = e\left(H_0(m), \mathsf{aggvk}\right),$$

then output 1 and otherwise output 0.

Lemma 3. *Let $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, p, g_1, g_2, e)$ as above, let A be an adversary, and let*

$$\epsilon = \mathsf{Adv}_{\Pi_{\mathsf{BLS}[\mathcal{G}]}}^{\mathsf{rMultiSig}}(A).$$

Then, either $\epsilon < \frac{2q_0 \cdot q_1^2}{p}$, or there exists a co-CDH algorithm $A_{\mathsf{co\text{-}CDH}}$ such that

$$\Pr_{\alpha, \beta \leftarrow \mathbb{Z}_p}\left[A_{\mathsf{co\text{-}CDH}}\left(\mathcal{G}, X\right) = g_1^{\alpha\beta}\right] \geq \frac{B}{8q_1} \cdot \left(\frac{\epsilon}{q_0}\right)^2$$

and

$$\mathbb{E}_{\alpha, \beta \leftarrow \mathbb{Z}_p}\left[\left(\mathsf{Time}\left(A_{\mathsf{co\text{-}CDH}}\left(\mathcal{G}, X\right)\right)\right)^2\right] \leq 2 \cdot (1 + B)^2 \cdot \left(t + q_0 \cdot \tau_{\mathsf{exp}_1} + q_{\mathsf{S}} \cdot \tau_{\mathsf{exp}_1} + \tau_{\mathsf{exp}_2^n}\right)^2 \cdot \frac{\epsilon}{q_0},$$

for $B = \left\lceil (q_0/\epsilon)^{1/2} - 1 \right\rceil$, where:

- $X = \left(g_1^{\alpha}, g_2^{\alpha}, g_2^{\beta}\right)$.

- *t is an upper bound on the running time of the adversary A.*
- *q_0, q_1 and q_{S} are upper bounds on the number of H_0-queries, H_1-queries and signing queries issued by the adversary A, respectively.*
- *n is an upper bound on the largest signer set involved in a single multi-signature during the experiment $\mathsf{Exp}^{\mathsf{rMultiSig}}_{\Pi_{\mathsf{BLS}[\mathcal{G}]}}(A)$.*
- *τ_{exp_1} is the time required to compute an exponentiation in $\mathbb{G}_1$, and $\tau_{\mathsf{exp}_2^n}$ is the time required to compute an n-multi-exponentiation in $\mathbb{G}_2$.*

Proof of Lemma 3. Let $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, p, g_1, g_2, e)$ as above, and let A be an adversary that runs in time at most t and issues at most q_{S}, q_0 and q_1 oracle queries to the signing oracle, H_0 and H_1, respectively. Without loss of generality, we assume that A does not query either of these oracles more than once with the same input (since all three are deterministic), and that when A outputs a potential forgery $\left(\vec{\mathsf{vk}}^*, m^*, \sigma^*\right)$ then it previously issued the H_1 query $\left(\mathsf{vk}^*, \vec{\mathsf{vk}}^*\right)$. For simplicity, we additionally assume that whenever A issues a signing query $\left(\vec{\mathsf{vk}}, m\right)$, then A previously issued the queries $H_0(m)$ and $H_1\left(\mathsf{vk}^*, \vec{\mathsf{vk}}\right)$ (this is not essential, and we discuss below how to avoid this assumption).

We transform the adversary A into an algorithm $\tilde{A}$ that is compatible with Lemma 1. The algorithm $\tilde{A}$ receives as input the group description $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, p, g_1, g_2, e)$, a triplet $X = \left(g_1^\alpha, g_1^\beta, g_2^\beta\right)$ of group elements, values $h_1, \ldots, h_{q_1} \in \mathbb{Z}_p$, as well as randomness $\rho_1, \rho_2 \in \{0,1\}^*$, where ρ_1 will be used as $\tilde{A}$'s own internal randomness and ρ_2 will be used as the randomness required for running A. The algorithm $\tilde{A}$ is defined as follows:

```
The Algorithm $\tilde{A}(g_1^\alpha, g_1^\beta, g_2^\beta, h_1, \ldots, h_{q_1}; \rho_1, \rho_2)$

1. Sample $k \leftarrow \{1, \ldots, q_0\}$, and let $\mathsf{vk}^* = g_2^\beta$.
2. Invoke $A(\mathsf{vk}^*; \rho_2)$ by responding to $A$'s oracle queries as follows:
   (a) For the $i$th $H_0$-query $m$:
       i. If $i = k$ then return $H_0(m) = g_1^\alpha$.
       ii. Else, sample $r_i \leftarrow \mathbb{Z}_p$ and return $H_0(m) = g_1^{r_i}$.
   (b) For the $j$th $H_1$-query $\left(\mathsf{vk}, \vec{\mathsf{vk}}\right)$:
       i. If $\mathsf{vk} = \mathsf{vk}^*$ and $\mathsf{vk} \in \vec{\mathsf{vk}}$ then return $H_1\left(\mathsf{vk}, \vec{\mathsf{vk}}\right) = h_j$.
       ii. Else, sample $H_1\left(\mathsf{vk}, \vec{\mathsf{vk}}\right) \leftarrow \mathbb{Z}_p$ and return it.
   (c) For any signing query $\left(\vec{\mathsf{vk}}, m\right)$:
       i. If $H_0(m) = g_1^\alpha$ then output $(0, \perp)$ and terminate.
       ii. Else, if $\mathsf{vk}^* \notin \vec{\mathsf{vk}}$ then return $\perp$.
       iii. Else, retrieve the value $r \in \mathbb{Z}_p$ previously chosen in Step 2(a)ii when responding to the query $H_0(m)$ for which $H_0(m) = g_1^r$, and the value $t = H_1\left(\mathsf{vk}, \vec{\mathsf{vk}}\right)$ previously chosen in Step 2(b)ii when responding to the query $H_1\left(\mathsf{vk}, \vec{\mathsf{vk}}\right)$, and return $\left(g_1^\beta\right)^{t \cdot r}$.
3. When $A$ outputs a forgery $\left(\vec{\mathsf{vk}}^*, m^*, \sigma^*\right)$, where $\vec{\mathsf{vk}}^* = (\mathsf{vk}_1^*, \ldots, \mathsf{vk}_n^*)$ for some $n \in \mathbb{N}$, proceed as follows:
   (a) Retrieve the value $a_j = H_1\left(\mathsf{vk}_j^*, \vec{\mathsf{vk}}^*\right)$ for every $j \in [n]$, and compute $\mathsf{aggvk}^* = \mathsf{KeyAgg}\left(\vec{\mathsf{vk}}^*\right)$ (if some of the $a_j$ values have not yet been defined, then sample them independently and uniformly).
   (b) If $H_0(m^*) \neq g_1^\alpha$ or $\mathsf{Verify}(\mathsf{aggvk}^*, m^*, \sigma^*) \neq 1$ then output $(0, \perp)$.
```

(c) Else, let $I \in [q_1]$ denote the index of the H_1-query $\left(\mathsf{vk}^*, \vec{\mathsf{vk}}^*\right)$ issued by A, and output (I, σ^*).

We observe that for uniformly and independently distributed $\alpha, \beta, h_1, \ldots, h_{q_1} \in \mathbb{Z}_p$ and $\rho_1, \rho_2 \in \{0,1\}^*$, the algorithm $\tilde{A}$ perfectly emulates the experiment $\mathsf{Adv}^{\mathsf{rMultiSig}}_{\Pi_{\mathsf{BLS}[\mathcal{G}]}}(A)$ to A, as long as A does not issue a signing query for the message m^*. Therefore, letting $\epsilon = \mathsf{Adv}^{\mathsf{rMultiSig}}_{\Pi_{\mathsf{BLS}[\mathcal{G}]}}(A)$, we obtain

$$\Pr_{\substack{\alpha,\beta,h_1,\ldots,h_{q_1} \leftarrow \mathbb{Z}_p \\ \rho_1,\rho_2 \leftarrow \{0,1\}^*}} \left[\tilde{A}\left(X, h_1, \ldots, h_{q_1}; \rho_1, \rho_2\right) = (I, \cdot) \text{ s.t. } I > 0\right] = \frac{\epsilon}{q_0},$$

where $X = \left(g_1^\alpha, g_1^\beta, g_2^\beta\right)$. In addition, in terms of $\tilde{A}$'s running time, note that it invokes A, and performs the following additional computations (as common, we focus on counting the additional number of group operations): (1) For each H_0-query $\tilde{A}$ computes at most one exponentiation in $\mathbb{G}_1$, (2) for each signing query $\tilde{A}$ computes at most one exponentiation in $\mathbb{G}_1$, and (3) when A provides an output $\tilde{A}$ computes an n-multi-exponentiation in $\mathbb{G}_2$ (for simplicity, we ignore an additional minor additive term due to the single pairing computed by $\tilde{A}$). Thus,

$$t_{\tilde{A}} \leq t + q_0 \cdot \tau_{\mathsf{exp}_1} + q_{\mathsf{S}} \cdot \tau_{\mathsf{exp}_1} + \tau_{\mathsf{exp}_2^n}$$

upper bounds the worst-case running time of $t_{\tilde{A}}$. Now, applying Lemma 1 for the algorithm $\tilde{A}$, $\mathcal{C} = \mathbb{Z}_p$, $q = q_1$ and $d = 2$, we obtain that either $\epsilon/q_0 \geq 2 \cdot q_1^2/p$, or that the algorithm $F_{\tilde{A},B}$ defined in Section 3 satisfies

$$\Pr_{\alpha,\beta \leftarrow \mathbb{Z}_p}\left[F_{\tilde{A},B}\left(\mathcal{G}, X\right) \neq \bot\right] \geq \frac{B}{8q_1} \cdot \left(\frac{\epsilon}{q_0}\right)^2$$

and

$$\mathbb{E}_{\alpha,\beta \leftarrow \mathbb{Z}_p}\left[\left(\mathsf{Time}\left(F_{\tilde{A},B}\left(\mathcal{G}, X\right)\right)\right)^2\right] \leq 2 \cdot (1+B)^2 \cdot \left(t + q_0 \cdot \tau_{\mathsf{exp}_1} + q_{\mathsf{S}} \cdot \tau_{\mathsf{exp}_1} + \tau_{\mathsf{exp}_2^n}\right)^2 \cdot \frac{\epsilon}{q_0},$$

where $B = \left\lceil (q_0/\epsilon)^{1/2} - 1\right\rceil$. Equipped with the algorithm F, consider the following co-CDH algorithm $A_{\mathsf{co\text{-}CDH}}$ that receives as input $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, p, g_1, g_2, e)$ and $X = \left(g_1^\alpha, g_1^\beta, g_2^\beta\right)$:

The Algorithm $A_{\mathsf{co\text{-}CDH}}\left(\mathcal{G}, X\right)$

1. Invoke $F_{\tilde{A},B}\left(\mathcal{G}, X\right)$ and output $\bot$ if $F_{\tilde{A},B}$ returns $\bot$.
2. Otherwise, denote by $\left(I, \sigma, \sigma', h_1, \ldots, h_{q_1}, h'_I, \ldots, h'_{q_1}\right)$ the output produced by $F_{\tilde{A},B}$, and output $(\sigma/\sigma')^{1/(h_I - h'_I)} \in \mathbb{G}_1$.

First, note that the distribution of $A_{\mathsf{co\text{-}CDH}}$'s running time is essentially identical to that of $F_{\tilde{A},B}$, where we ignore for simplicity the additional minor additive term due to the computation of its output $(\sigma/\sigma')^{1/(h_I - h'_I)}$.

Second, note that whenever $F_{\tilde{A},B}\left(\mathcal{G}, X\right)$ outputs $\left(I, \sigma, \sigma', h_1, \ldots, h_{q_1}, h'_I, \ldots, h'_{q_1}\right)$, then for some $\rho_1, \rho_2 \in \{0,1\}^*$ it holds that

$$\begin{aligned}
(I, \sigma) &= \tilde{A}\left(g_1^\alpha, g_1^\beta, g_2^\beta, h_1, \ldots, h_{q_1}; \rho_1, \rho_2\right) \\
(I, \sigma') &= \tilde{A}\left(g_1^\alpha, g_1^\beta, g_2^\beta, h_1, \ldots, h_{I-1}, h'_I, \ldots, h'_{q_1}; \rho_1, \rho_2\right),
\end{aligned}$$

where $h_I \neq h'_I$. These two executions are identical up until $\tilde{A}$ responds to the I-th H_1-query issued by A, and therefore in both executions the I-th H_1-query issued by A is the same $\left(\mathsf{vk}^*, \vec{\mathsf{vk}}^*\right)$. Moreover, this also implies that in both executions A produces a forgery with respect to the same vk^* and $\vec{\mathsf{vk}}^*$, where in the first execution it holds that $H_1\big(\mathsf{vk}^*, \vec{\mathsf{vk}}^*\big) = h_I$ and in the second execution it holds that $H_1\big(\mathsf{vk}^*, \vec{\mathsf{vk}}^*\big) = h'_I$. Letting $\vec{\mathsf{vk}}^* = \big(\mathsf{vk}_1^*, \dots, \mathsf{vk}_n^*\big)$, and denoting by $\ell \in [n]$ the index for which $\mathsf{vk}^* = \mathsf{vk}_\ell^*$, we observe that in both execution the values $H_1\big(\mathsf{vk}_i^*, \vec{\mathsf{vk}}^*\big)$ are identical for every $i \neq \ell$ since these values are sampled the same internal randomness ρ_1.

The fact that A produces valid signatures σ and σ' in these two executions implies that

$$e(\sigma, g_2) = e\left(g_1^\alpha, (\mathsf{vk}^*)^{h_I} \cdot \prod_{i \in n \setminus \{\ell\}} (\mathsf{vk}_i^*)^{H_1\left(\mathsf{vk}_i^*, \vec{\mathsf{vk}}^*\right)}\right)$$

$$e(\sigma', g_2) = e\left(g_1^\alpha, (\mathsf{vk}^*)^{h'_I} \cdot \prod_{i \in n \setminus \{\ell\}} (\mathsf{vk}_i^*)^{H_1\left(\mathsf{vk}_i^*, \vec{\mathsf{vk}}^*\right)}\right),$$

and therefore

$$e(\sigma/\sigma', g_2) = e\left(g_1^\alpha, (\mathsf{vk}^*)^{h_I - h'_I}\right) = e\left(g_1^\alpha, \left(g_2^\beta\right)^{h_I - h'_I}\right) = e\left(g_1^{\alpha\beta}, g_2\right)^{h_I - h'_I}.$$

Thus, $(\sigma/\sigma')^{1/(h_I - h'_I)} = g_1^{\alpha\beta}$, and we obtain

$$\Pr_{\alpha,\beta \leftarrow \mathbb{Z}_p}\left[A_{\mathsf{co\text{-}CDH}}(\mathcal{G}, X) = g_1^{\alpha\beta}\right] \geq \Pr_{\alpha,\beta \leftarrow \mathbb{Z}_p}\left[F_{\tilde{A},B}(\mathcal{G}, X) \neq \bot\right] \geq \frac{B}{8q_1} \cdot \left(\frac{\epsilon}{q_0}\right)^2$$

as required. ■

Recall that when proving Lemma 3, we have additionally assumed that whenever A issues a signing query $\big(\vec{\mathsf{vk}}, m\big)$, then A previously issued the queries $H_0(m)$ and $H_1\big(\mathsf{vk}^*, \vec{\mathsf{vk}}\big)$. The simplest way to avoid this assumption is to transform A into a completely equivalent algorithm that issues these two queries upon any signing query. This would increase the number of H_0-queries and H_1-queries issued by A by an additive q_{S} term, and thus have a minor effect on the resulting concrete security bound. However, as for the number of H_0 queries, this is not essential. Specifically, for avoiding a (potential) additional query $H_0(m)$ with each signing query, we can modify $\tilde{A}$ as follows: Upon any signing query $\big(\vec{\mathsf{vk}}, m\big)$, since it is guaranteed that $m \neq m^*$ (recall that we are considering the relaxed experiment as discussed following the statement of Theorem 7), then $\tilde{A}$ can first execute Step 2(a)ii for sampling a value $r \leftarrow \mathbb{Z}_p$ and defining $H_0(m) = g_1^r$ before proceeding to Step 2(c)iii.

Equipped with Lemma 3, we can now derive the proof of Theorem 7.

Proof of Theorem 7. Let $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, p, g_1, g_2, e)$ as above, let A be an adversary, and let

$$\epsilon = \mathsf{Adv}^{\mathsf{rMultiSig}}_{\Pi_{\mathsf{BLS}[\mathcal{G}]}}(A).$$

Lemma 3 states that either $\epsilon < 2q_0 \cdot q_1^2/p$, or there exists a co-CDH algorithm $A_{\mathsf{co\text{-}CDH}}$ such that

$$\Pr_{\alpha,\beta \leftarrow \mathbb{Z}_p}\left[A_{\mathsf{co\text{-}CDH}}\left(\mathcal{G}, g_1^\alpha, g_1^\beta, g_2^\beta\right) = g_1^{\alpha\beta}\right] \geq \frac{B}{8q_1} \cdot \left(\frac{\epsilon}{q_0}\right)^2$$

and

$$\mathbb{E}_{\alpha,\beta \leftarrow \mathbb{Z}_p}\left[\left(\mathsf{Time}\left(A_{\mathsf{co\text{-}CDH}}\left(\mathcal{G}, g_1^\alpha, g_1^\beta, g_2^\beta\right)\right)\right)^2\right]$$

$$\leq 2 \cdot (1+B)^2 \cdot \left(t + q_0 \cdot \tau_{\mathsf{exp}_1} + q_{\mathsf{S}} \cdot \tau_{\mathsf{exp}_1} + \tau_{\mathsf{exp}_2^n}\right)^2 \cdot \frac{\epsilon}{q_0},$$

for $B = \left\lceil (q_0/\epsilon)^{1/2} - 1 \right\rceil$. Assuming the co-CDH problem is second-moment hard in $\mathcal{G}$ guarantees that

$$\Pr_{\alpha,\beta \leftarrow \mathbb{Z}_p}\left[A_{\textsf{co-CDH}}\left(\mathcal{G}, g_1^{\alpha}, g_1^{\beta}, g_2^{\beta}\right) = g_1^{\alpha\beta} \right] \leq \frac{\mathbb{E}_{\alpha,\beta \leftarrow \mathbb{Z}_p}\left[\left(\textsf{Time}\left(A_{\textsf{co-CDH}}\left(\mathcal{G}, g_1^{\alpha}, g_1^{\beta}, g_2^{\beta}\right)\right)\right)^2 \right]}{p},$$

and thus

$$\frac{B}{8q_1} \cdot \left(\frac{\epsilon}{q_0}\right)^2 \leq \frac{2 \cdot (1+B)^2 \cdot \left(t + q_0 \cdot \tau_{\textsf{exp}_1} + q_{\textsf{S}} \cdot \tau_{\textsf{exp}_1} + \tau_{\textsf{exp}_2^n}\right)^2}{p} \cdot \frac{\epsilon}{q_0}.$$

This implies that

$$\epsilon \leq \left(16 q_0^{3/2} \cdot q_1 \cdot \frac{\left(t + q_0 \cdot \tau_{\textsf{exp}_1} + q_{\textsf{S}} \cdot \tau_{\textsf{exp}_1} + \tau_{\textsf{exp}_2^n}\right)^2}{p} \right)^{2/3}.$$

Taking into account also the case in which $\epsilon < 2q_0 \cdot q_1^2/p$ and using the fact that $q_1 \leq t$, we obtain

$$\begin{aligned} \epsilon &\leq \max\left\{ \left(16 q_0^{3/2} \cdot q_1 \cdot \frac{\left(t + q_0 \cdot \tau_{\textsf{exp}_1} + q_{\textsf{S}} \cdot \tau_{\textsf{exp}_1} + \tau_{\textsf{exp}_2^n}\right)^2}{p} \right)^{2/3}, \frac{2q_0 \cdot q_1^2}{p} \right\} \\ &= \left(16 q_0^{3/2} \cdot q_1 \cdot \frac{\left(t + q_0 \cdot \tau_{\textsf{exp}_1} + q_{\textsf{S}} \cdot \tau_{\textsf{exp}_1} + \tau_{\textsf{exp}_2^n}\right)^2}{p} \right)^{2/3}, \end{aligned}$$

which settles the proof of Theorem 7. ■

References

[AABN02] Michel Abdalla, Jee Hea An, Mihir Bellare, and Chanathip Namprempre. From identification to signatures via the Fiat-Shamir transform: Minimizing assumptions for security and forward-security. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 418–433. Springer, Heidelberg, April / May 2002. doi:10.1007/3-540-46035-7_28.

[AB21] Handan Kilinç Alper and Jeffrey Burdges. Two-round trip Schnorr multi-signatures via delinearized witnesses. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021, Part I*, volume 12825 of *Lecture Notes in Computer Science*, pages 157–188, Virtual Event, August 2021. Springer, Heidelberg. doi:10.1007/978-3-030-84242-0_7.

[AHK20] Thomas Agrikola, Dennis Hofheinz, and Julia Kastner. On instantiating the algebraic group model from falsifiable assumptions. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 96–126. Springer, Heidelberg, May 2020. doi:10.1007/978-3-030-45724-2_4.

[BB08] Dan Boneh and Xavier Boyen. Short signatures without random oracles and the SDH assumption in bilinear groups. *Journal of Cryptology*, 21(2):149–177, April 2008. doi:10.1007/s00145-007-9005-7.

[BCC$^+$16] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 327–357. Springer, Heidelberg, May 2016. doi:10.1007/978-3-662-49896-5_12.

[BD20] Mihir Bellare and Wei Dai. The multi-base discrete logarithm problem: Tight reductions and non-rewinding proofs for Schnorr identification and signatures. In Karthikeyan Bhargavan, Elisabeth Oswald, and Manoj Prabhakaran, editors, *Progress in Cryptology - INDOCRYPT 2020: 21st International Conference in Cryptology in India*, volume 12578 of *Lecture Notes in Computer Science*, pages 529–552. Springer, Heidelberg, December 2020. doi:10.1007/978-3-030-65277-7_24.

[BD21] Mihir Bellare and Wei Dai. Chain reductions for multi-signatures and the HBMS scheme. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021, Part IV*, volume 13093 of *Lecture Notes in Computer Science*, pages 650–678. Springer, Heidelberg, December 2021. doi:10.1007/978-3-030-92068-5_22.

[BDN18] Dan Boneh, Manu Drijvers, and Gregory Neven. Compact multi-signatures for smaller blockchains. In Thomas Peyrin and Steven Galbraith, editors, *Advances in Cryptology – ASIACRYPT 2018, Part II*, volume 11273 of *Lecture Notes in Computer Science*, pages 435–464. Springer, Heidelberg, December 2018. doi:10.1007/978-3-030-03329-3_15.

[BFL20] Balthazar Bauer, Georg Fuchsbauer, and Julian Loss. A classification of computational assumptions in the algebraic group model. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020, Part II*, volume 12171 of *Lecture Notes in Computer Science*, pages 121–151. Springer, Heidelberg, August 2020. doi:10.1007/978-3-030-56880-1_5.

[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: 14th Conference on Computer and Communications Security*, pages 276–285. ACM Press, October 2007. doi:10.1145/1315245.1315280.

[BLS01] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. In Colin Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532. Springer, Heidelberg, December 2001. doi:10.1007/3-540-45682-1_30.

[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: 13th Conference on Computer and Communications Security*, pages 390–399. ACM Press, October / November 2006. doi:10.1145/1180405.1180453.

[Bol03] Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In Yvo Desmedt, editor, *PKC 2003: 6th International Workshop on Theory and Practice in Public Key Cryptography*, volume 2567 of *Lecture Notes in Computer Science*, pages 31–46. Springer, Heidelberg, January 2003. doi:10.1007/3-540-36288-6_3.

[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: 1st Conference on Computer and Communications Security*, pages 62–73. ACM Press, November 1993. doi:10.1145/168588.168596.

[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, *Advances in Cryptology – CRYPTO 2022, Part II*, volume 13508 of *Lecture Notes in Computer Science*, pages 276–305. Springer, Heidelberg, August 2022. doi:10.1007/978-3-031-15979-4_10.

[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. doi:10.1109/SP.2019.00050.

[DOTT22] Ivan Damgård, Claudio Orlandi, Akira Takahashi, and Mehdi Tibouchi. Two-round n-out-of-n and multi-signatures and trapdoor commitment from lattices. *Journal of Cryptology*, 35(2):14, April 2022. doi:10.1007/s00145-022-09425-3.

[FKL18] Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 33–62. Springer, Heidelberg, August 2018. doi:10.1007/978-3-319-96881-0_2.

[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, *Advances in Cryptology – EUROCRYPT 2020, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 63–95. Springer, Heidelberg, May 2020. doi:10.1007/978-3-030-45724-2_3.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology – CRYPTO'86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, Heidelberg, August 1987. doi:10.1007/3-540-47721-7_12.

[FSZ22] Nils Fleischhacker, Mark Simkin, and Zhenfei Zhang. Squirrel: Efficient synchronized multi-signatures from lattices. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022: 29th Conference on Computer and Communications Security*, pages 1109–1123. ACM Press, November 2022. doi:10.1145/3548606.3560655.

[IN83] Kazuharu Itakura and Katsuhiro Nakamura. A public-key cryptosystem suitable for digital multisignatures. *NEC Research & Development*, 71:1–8, 1983.

[JT20] Joseph Jaeger and Stefano Tessaro. Expected-time cryptography: Generic techniques and applications to concrete soundness. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020: 18th Theory of Cryptography Conference, Part III*, volume 12552 of *Lecture Notes in Computer Science*, pages 414–443. Springer, Heidelberg, November 2020. doi:10.1007/978-3-030-64381-2_15.

[KMP16] Eike Kiltz, Daniel Masny, and Jiaxin Pan. Optimal security proofs for signatures from identification schemes. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 33–61. Springer, Heidelberg, August 2016. doi:10.1007/978-3-662-53008-5_2.

[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, *Advances in Cryptology – EUROCRYPT'94*, volume 950 of *Lecture Notes in Computer Science*, pages 194–204. Springer, Heidelberg, May 1995. doi:10.1007/BFb0053435.

[LK23] Kwangsu Lee and Hyoseung Kim. Two-round multi-signatures from Okamoto signatures. *Mathematics*, 11(14), 2023. doi:10.3390/math11143223.

[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, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 465–485. Springer, Heidelberg, May / June 2006. doi:10.1007/11761679_28.

[Mau05] Ueli M. Maurer. Abstract models of computation in cryptography (invited paper). In Nigel P. Smart, editor, *10th IMA International Conference on Cryptography and Coding*, volume 3796 of *Lecture Notes in Computer Science*, pages 1–12. Springer, Heidelberg, December 2005. doi:10.1007/11586821_1.

[MOR01] Silvio Micali, Kazuo Ohta, and Leonid Reyzin. Accountable-subgroup multisignatures: Extended abstract. In Michael K. Reiter and Pierangela Samarati, editors, *ACM CCS 2001: 8th Conference on Computer and Communications Security*, pages 245–254. ACM Press, November 2001. doi:10.1145/501983.502017.

[MPSW19] Gregory Maxwell, Andrew Poelstra, Yannick Seurin, and Pieter Wuille. Simple schnorr multi-signatures with applications to Bitcoin. *Designs, Codes and Cryptography*, 87(9):2139–2164, 2019. doi:10.1007/s10623-019-00608-x.

[MTT19] Taiga Mizuide, Atsushi Takayasu, and Tsuyoshi Takagi. Tight reductions for Diffie-Hellman variants in the algebraic group model. In Mitsuru Matsui, editor, *Topics in Cryptology – CT-RSA 2019*, volume 11405 of *Lecture Notes in Computer Science*, pages 169–188. Springer, Heidelberg, March 2019. doi:10.1007/978-3-030-12612-4_9.

[NRS21] Jonas Nick, Tim Ruffing, and Yannick Seurin. MuSig2: Simple two-round Schnorr multi-signatures. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021, Part I*, volume 12825 of *Lecture Notes in Computer Science*, pages 189–221, Virtual Event, August 2021. Springer, Heidelberg. doi:10.1007/978-3-030-84242-0_8.

[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: 27th Conference on Computer and Communications Security*, pages 1717–1731. ACM Press, November 2020. doi:10.1145/3372297.3417236.

[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, *Advances in Cryptology – ASIACRYPT'91*, volume 739 of *Lecture Notes in Computer Science*, pages 139–148. Springer, Heidelberg, November 1993. doi:10.1007/3-540-57332-1_11.

[PS00] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, June 2000. doi:10.1007/s001450010003.

[PW23] Jiaxin Pan and Benedikt Wagner. Chopsticks: Fork-free two-round multi-signatures from non-interactive assumptions. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part V*, volume 14008 of *Lecture Notes in Computer Science*, pages 597–627. Springer, Heidelberg, April 2023. doi:10.1007/978-3-031-30589-4_21.

[RS20] Lior Rotem and Gil Segev. Algebraic distinguishers: From discrete logarithms to decisional Uber assumptions. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020: 18th Theory of Cryptography Conference, Part III*, volume 12552 of *Lecture Notes in Computer Science*, pages 366–389. Springer, Heidelberg, November 2020. doi:10.1007/978-3-030-64381-2_13.

[RS21] Lior Rotem and Gil Segev. Tighter security for Schnorr identification and signatures: A high-moment forking lemma for Σ-protocols. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021, Part I*, volume 12825 of *Lecture Notes in Computer Science*, pages 222–250, Virtual Event, August 2021. Springer, Heidelberg. doi:10.1007/978-3-030-84242-0_9.

[RY07] Thomas Ristenpart and Scott Yilek. The power of proofs-of-possession: Securing multiparty signatures against rogue-key attacks. In Moni Naor, editor, *Advances in Cryptology – EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 228–245. Springer, Heidelberg, May 2007. doi:10.1007/978-3-540-72540-4_13.

[Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *Advances in Cryptology – EUROCRYPT'97*, volume 1233 of *Lecture Notes in Computer Science*, pages 256–266. Springer, Heidelberg, May 1997. doi:10.1007/3-540-69053-0_18.

[SSY23] Gil Segev, Amit Sharabi, and Eylon Yogev. Rogue-instance security for batch knowledge proofs. In Guy N. Rothblum and Hoeteck Wee, editors, *TCC 2023: 21st Theory of Cryptography Conference, Part I*, volume 14369 of *Lecture Notes in Computer Science*, pages 121–157. Springer, Heidelberg, November / December 2023. doi:10.1007/978-3-031-48615-9_5.

[TZ23] Stefano Tessaro and Chenzhi Zhu. Threshold and multi-signature schemes from linear hash functions. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part V*, volume 14008 of *Lecture Notes in Computer Science*, pages 628–658. Springer, Heidelberg, April 2023. doi:10.1007/978-3-031-30589-4_22.

A Proofs of Claims 3 and 4

A.1 Proof of Claim 3

Consider the functions $f, g : [0,1] \to \mathbb{R}$ defined as follows:

$$f(z) = z \cdot (1 - (1-z)^B)$$
$$g(z) = \begin{cases} \frac{1}{2} \cdot B \cdot z^2 & \text{if } 0 \leq z < \frac{1}{B} \\ z - \frac{1}{2 \cdot B} & \text{otherwise} \end{cases}$$

Then,

$$\begin{aligned}\mathbb{E}\left[\tilde{\epsilon_i}(x,\rho,\vec{h}_{i-1}) \cdot \left(1 - \left(1 - \tilde{\epsilon_i}(x,\rho,\vec{h}_{i-1})\right)\right)^B\right] &= \mathbb{E}\left[f(\tilde{\epsilon_i}(x,\rho,\vec{h}_{i-1}))\right] \\ &\geq \mathbb{E}\left[g(\tilde{\epsilon_i}(x,\rho,\vec{h}_{i-1}))\right] \\ &\geq g\left(\mathbb{E}\left[\tilde{\epsilon_i}(x,\rho,\vec{h}_{i-1})\right]\right)\end{aligned}$$

Recall that we have defined $\epsilon_i = \Pr\left[I_0 = i\right]$ and $\tilde{\epsilon_i} = \mathbb{E}\big[\tilde{\epsilon_i}(x,\rho,\vec{h}_{i-1})\big]$, and therefore

$$\tilde{\epsilon_i} \geq \max\left\{0, \Pr\left[I_0 = i\right] - \frac{q}{|\mathcal{C}|}\right\} = \max\left\{0, \epsilon_i - \frac{q}{|\mathcal{C}|}\right\}$$

Since for every $i \in [q]$ it holds that $\epsilon_i \leq \epsilon \leq \frac{1}{B}$, then we obtain $\tilde{\epsilon_i} \leq \epsilon \leq \frac{1}{B}$. Hence, by the definition of g we obtain

$$\begin{aligned}\mathbb{E}\left[\tilde{\epsilon_i}(x,\rho,\vec{h}_{i-1}) \cdot \left(1 - \left(1 - \tilde{\epsilon_i}(x,\rho,\vec{h}_{i-1})\right)\right)^B\right] &\geq g\left(\mathbb{E}\left[\tilde{\epsilon_i}(x,\rho,\vec{h}_{i-1})\right]\right) \\ &= \frac{1}{2} \cdot B \cdot \tilde{\epsilon_i}^2\end{aligned}$$

■

A.2 Proof of Claim 4

By definition, it holds that

$$\tilde{\epsilon_i} = \mathbb{E}_{x,\rho,\vec{h}_{i-1}}\left[\epsilon_i(x,\rho,\vec{h}_{i-1})\right] = \mathbb{E}_{x,\rho,\vec{h}_{i-1}}\left[\max\left\{0, \Pr_{h_i,\ldots,h_q}\left[I_0 = i \middle| x,\rho,\vec{h}_{i-1}\right] - \frac{q}{|\mathcal{C}|}\right\}\right].$$

Therefore,

$$\begin{aligned}&\mathbb{E}_{x,\rho,\vec{h}_{i-1}}\left[\max\left\{0, \Pr_{h_i,\ldots,h_q}\left[I_0 = i \middle| x,\rho,\vec{h}_{i-1}\right] - \frac{q}{|\mathcal{C}|}\right\}\right] \tag{7} \\ &\quad\geq \mathbb{E}_{x,\rho,\vec{h}_{i-1}}\left[\Pr_{h_i,\ldots,h_q}\left[I_0 = i \middle| x,\rho,\vec{h}_{i-1}\right] - \frac{q}{|\mathcal{C}|}\right] \tag{8} \\ &\quad= \mathbb{E}_{x,\rho,\vec{h}_{i-1}}\left[\Pr_{h_i,\ldots,h_q}\left[I_0 = i \middle| x,\rho,\vec{h}_{i-1}\right]\right] - \frac{q}{|\mathcal{C}|} \\ &\quad= \sum_{x,\rho,\vec{h}_{i-1}}\left(\Pr\left[x,\rho,\vec{h}_{i-1}\right] \cdot \Pr_{h_i,\ldots,h_q}\left[I_0 = i\right]\right) - \frac{q}{|\mathcal{C}|} \\ &\quad= \sum_{x,\rho,\vec{h}_{i-1}}\left(\Pr\left[x,\rho,\vec{h}_{i-1}\right] \cdot \Pr_{h_i,\ldots,h_q}\left[I_0 = i \middle| x,\rho,\vec{h}_{i-1} \middle| x,\rho,\vec{h}_{i-1}\right]\right) - \frac{q}{|\mathcal{C}|} \\ &\quad= \Pr_{x,\rho,\vec{h}_{i-1},h_i,\ldots,h_q}\left[I_i = i \middle| x,\rho,\vec{h}_{i-1}\right] - \frac{q}{|\mathcal{C}|} \tag{9} \\ &\quad= \epsilon_i - \frac{q}{|\mathcal{C}|}\end{aligned}$$

where Eq. (8) follows from the monotonicity of the expectation, and Eq. (9) follows from the law of total probability. ■