

On Sparkle+ and its Security Claims

Andrea Gangemi¹, Massimiliano Sala^{1,2}, and Lorenzo Viganò²

¹ ShunyaXifra

agangemi@shunyx.net

² University of Trento, Department of Mathematics, Via Sommarive 14, Povo, Trento,
38123, Italy

massimiliano.sala@unitn.it, lorenzo.vigano@studenti.unitn.it

Abstract. We revisit the security bounds of **Sparkle+**, the threshold Schnorr signature scheme which first appeared in 2023. This scheme has been updated and corrected in several versions and papers (also of other authors). However, no version or paper has checked in detail the final bounds of Theorem 1 and 2 of the June 2025 version, which have therefore, to the best of our knowledge, been taken for granted in the related literature. There, the derivation of the final bounds of its Theorems 1 and 2, on full static security under DL and on adaptive security up to $t/2$ corruptions under AOMDL, from the intermediate inequalities established by their proofs is left to the reader. Carrying it out, we obtain bounds that differ from the stated ones in three distinct algebraic respects: the terms produced by the two game hops precede the rewinding and therefore appear outside the square root rather than inside it; the whole game-hop aggregate, and not only its collision part, re-enters through the extraction inequality with coefficient $2q$ rather than 1; and the exact inversion of the general forking lemma contributes $q/2p$ outside the root together with $q^2/4p^2$ inside it, with q the number of queries and p the group order, its standard relaxation combining the two into an additive q/p , and the statements carry none of this. We prove the bounds that do follow, for both theorems, and determine exactly when each of ours and the original is the tighter. Conditional on the correctness of the game transitions from which those inequalities come, the asymptotic security conclusions are unaffected.

Keywords: Threshold Signature Schemes · Static Security · Adaptive Security

1 Introduction

A threshold signature scheme (TSS) splits the ability to sign among a set of n parties, in a way that any subset of at least $t + 1$ parties can jointly produce a valid signature, while no coalition of at most t parties can reconstruct the signing key or forge a signature, under the security notion of Section 3. Although the generic notion admits any qualified set, the analysis of **Sparkle+** considered here fixes every signing set to exactly $t + 1$ parties. Usually, signing is performed via an

interactive protocol whose output is a signature under the joint public key, and in schemes such as **Sparkle+** the protocol outputs an ordinary Schnorr signature under the joint public key, so that verification is identical to single-party Schnorr verification and the process is transparent for the verifier.

The concept goes back to the foundational work of Desmedt and Frankel [20], and has recently seen a surge of renewed interest, driven by applications such as distributed custody of cryptocurrency wallets [21] and by ongoing standardization efforts, such as the NIST First Call for Multi-Party Threshold Schemes [8]. By avoiding the storage of the complete signing key on a single device, threshold signatures remove a major single point of compromise.

Static and Adaptive Security. The security of a TSS is defined with respect to an adversary who can corrupt up to t of the n signers. A fundamental distinction concerns *when* the adversary must commit to the set of parties it will corrupt. In the *static* model, the adversary must declare the set of corrupted parties before the keys are generated, and this set remains fixed throughout the security experiment. This is the most common security model, but it imposes an artificial constraint on the actions the attacker can take. In the *adaptive* model, the adversary can corrupt parties dynamically, choosing the next one on the basis of everything it has observed so far. For a fixed corruption budget and state-exposure model, adaptive security implies the corresponding static guarantee and is therefore the stronger notion; it has proven notably hard to achieve.

It has long been known that there are schemes that are statically secure yet not adaptively secure [9]. One of the first papers to introduce adaptive security into the threshold setting is the work of Canetti, Gennaro, Jarecki, Krawczyk and Rabin [10], who proposed a generic reduction from adaptive to static security consisting in guessing the corrupted set in advance. This technique entails a security loss of $\binom{n}{t}$, which is exponential when $t = \Theta(n)$ and in any case destroys any meaningful concrete guarantee in the large-threshold regimes of interest. Proving adaptive security directly is challenging because the reduction must produce, on demand, consistent internal states for parties it did not anticipate corrupting; standard techniques for doing so often fail or lead to prohibitive losses, which explains why many modern threshold schemes were first analysed only in the static model.

Sparkle and Sparkle+. Against this background, Crites, Komlo and Maller introduced a three-round threshold Schnorr scheme together with new adaptive-security claims. The scheme introduced in the CRYPTO 2023 proceedings version [19] is a three-round threshold Schnorr signature called **Sparkle**, following a commit-reveal paradigm. Subsequently, the authors of **Twinkle** [4] observed that the analysis did not account for executions in which honest parties hold inconsistent views of the commitments broadcast in the first signing round, a situation that can arise already under static corruptions. The fix adopted in the full version [14] is to have each party sign its own view of the protocol using an auxiliary EUF-CMA-secure signature scheme **DS**; the resulting scheme is called **Sparkle+**. Three security properties are claimed for it:

1. *full static security*, that is static security with up to t corrupted parties, under the discrete logarithm (DL) assumption and the EUF-CMA security of DS, with the two hash functions of Sparkle+ modelled as random oracles (ROM); the unforgeability of DS is a property of DS alone, and only the proposed Schnorr instantiation brings a random oracle of its own;
2. adaptive security with up to $\lfloor t/2 \rfloor$ corrupted parties, under the *Algebraic One-More Discrete Logarithm* (AOMDL) assumption and the EUF-CMA security of DS, in the ROM;
3. *full adaptive security*, that is adaptive security against up to t corrupted parties, under AOMDL and the EUF-CMA security of DS, in the *algebraic group model* (AGM) and the ROM.

It is worth stressing at the outset that Sparkle+, and with it the terms involving $\text{Adv}^{\text{EUF-CMA}}$ in the bounds of the first two items, exist only in the full version: they are an artefact of the repair just described, and are absent from the proceedings version.

Prior Security Analyses. The third of the three claims was subsequently called into question. Crites and Stewart [18] present a plausible attack on a broad family of threshold Schnorr schemes with polynomial shares, Sparkle+ among them, and argue that full adaptive security requires either modifying the schemes or assuming the hardness of a new search problem that they introduce. A complementary result is obtained in [13], where the authors show that, for a broad class of threshold Schnorr schemes, proving adaptive security requires a new assumption, which they call *Low-Dimensional Vector Representation* (LDVR); in the same paper FROST [22] is proven fully adaptively secure under LDVR and AOMDL in the AGM and ROM. By metareduction techniques, [12] establishes impossibility results for broad classes of reductions proving the adaptive security of *key-unique* threshold signatures: for non-interactive computational and decisional assumptions, and, for natural rewinding reductions, for several one-more assumptions in specified corruption ranges. More recently, Acharya, Fuchsbauer, O’Neill and Seiffrank [1] gave a tight proof of full adaptive security in the pure ROM, without the AGM, under a new *Vandermonde Circular Discrete Logarithm* assumption; their new static and full-adaptive security theorems concern the original Sparkle, for which they also give the first proof of static security, rather than Sparkle+. Their introduction also surveys the wave of low-round adaptively secure threshold Schnorr schemes proposed in the meantime, among them Rackle [23], three-round and secure under DL in the pure ROM, Gargos [3], three-round and secure under DDH in the ROM, and the masked variants ms-FROST [2] and FaFROST [5] of FROST.

The first two claims have also been revisited, in a way that bears directly on the present note. Cho, Fuchsbauer, O’Neill and Seiffrank [11] give a completely tight reduction from the static security of Sparkle+ to the circular discrete-logarithm (CDL) assumption, giving a numerically far tighter bound, under the significantly stronger and newer CDL assumption, and replacing the factor n of the DS term by multi-user security of DS. In the course of that work they identify

an inconsistency in the static simulation of [14], namely that the random oracle H_{cm} is programmed for all honest parties of the signing set at once, whereas only one of the corresponding commitments originates from a previous first-round query. The June 2025 revision of [14] repairs this by programming H_{cm} only gradually, buffering the remaining values until they are explicitly queried, and it propagates the repair to the $t/2$ adaptive proof, at the cost of two additional bad events. All statements we discuss below refer to that revision.

Our Contribution. We examine the full static security and the $t/2$ adaptive security of **Sparkle+**, as proven in the appendices of [14], where the final bounds are not derived, but they are supposed to come from combining a list of intermediate inequalities.

By writing the full mathematical derivation, we obtain bounds that differ significantly from the stated ones. More precisely:

- we prove the resulting bounds for both theorems, Theorem 17

$$\text{Adv}_{\mathcal{A}, \text{Sparkle}^+}^{\text{TS-EUF-CMA}}(\kappa) \leq \sqrt{\frac{q^2}{4p^2} + A + 2qB} + \frac{q}{2p} + B + C,$$

where

$$A = q \cdot \text{Adv}_{B_2}^{\text{DL}}(\kappa), \quad B = \frac{q^2}{2p} + \frac{2q}{p}, \quad C = n \cdot \text{Adv}_{B_1, \text{DS}}^{\text{EUF-CMA}}(\kappa),$$

and Theorem 22

$$\text{Adv}_{\mathcal{A}, \text{Sparkle}^+}^{\text{adp-TS-EUF-CMA}}(\kappa, \frac{1}{2}) \leq \sqrt{\frac{q^2}{4p^2} + A' + 2qB'} + \frac{q}{2p} + B' + C',$$

where

$$A' = q \cdot \text{Adv}_{B'_2}^{t+1\text{-AOMDL}}(\kappa), \quad B' = \frac{q^2}{2p} + \frac{5q}{p}, \quad C' = \frac{n^2}{n - t/2} \cdot \text{Adv}_{B_1, \text{DS}}^{\text{EUF-CMA}}(\kappa);$$

- we make the identification of $\Pr[W_2]$ with the acceptance probability of the simulator explicit in the static proof and give its adaptive analogue;
- we derive exact criteria for comparing our bounds, in both their exact and their relaxed form, with the stated ones, together with simple sufficient conditions;
- we also show that the stated bound does not follow from the inequalities its proof supplies.

Under the models of Section 7.3, and for the instantiation of DS by a Schnorr signature proposed in [14], our bound is tighter by between twelve and thirty-nine bits at the parameter sets considered there.

Remark. *The asymptotic security conclusions are unaffected: conditional on the correctness of the underlying game transitions and reductions, which we take as given, the revised bounds remain negligible under the same assumptions and polynomial resource bounds.*

Remark. *We do not claim that the bounds studied here are the numerically strongest available for Sparkle+: [11] gives a tight bound for the same scheme and the same corruption model, but under the circular discrete-logarithm assumption, which is stronger and newer than the discrete logarithm assumption used here. The comparison is therefore across assumptions and not within one. To the best of our knowledge the statements we examine remain the only ones for Sparkle+ under the standard assumption, so their concrete value is what a reader unwilling to assume CDL has to work with.*

Structure of the Paper. Section 2 recalls the features needed to understand the structure of the Sparkle+ protocol, and Section 3 the assumptions and security notions on which its analysis rests. Section 4 isolates the forking step: we record the two formulations of the general forking lemma used in [14] and the exact inversion that links them, which is the step the security proofs require. Section 5 recalls the full static security theorem, lists the intermediate inequalities its proof establishes, and carries out the combination those inequalities support; we then compare the result with the stated bound. Section 6 does the same for adaptive security with $t/2$ corrupted parties. Section 7 then determines, in both settings, the parameter region in which each of the two bounds is the tighter, and instantiates the criterion at concrete parameters. Section 8 records how the stated bounds have changed across the four versions of [14]. Finally, Section 9 summarises the conclusions and identifies directions for future work.

2 Preliminaries

2.1 Notation

Let $\mathbb{G}$ denote a finite cyclic group with generator g and prime order p . With κ we denote the security parameter of the scheme or game involved. For a non-empty set S , let $x \xleftarrow{\$} S$ denote the sampling of an element from S uniformly at random; we use the same arrow for the output of a randomised algorithm, writing $y \xleftarrow{\$} \mathcal{A}(x)$ for the result of running $\mathcal{A}$ on x with uniformly chosen coins. A function $\nu : \mathbb{N} \rightarrow \mathbb{R}$ is called negligible if for all $c \in \mathbb{R}$, $c > 0$, there exists $k_0 \in \mathbb{N}$ such that $|\nu(k)| < \frac{1}{k^c}$ for all $k \in \mathbb{N}$, $k \geq k_0$. We refer to probabilistic polynomial-time adversaries with the initials PPT. Throughout, n , t and p satisfy $1 \leq t < n < p$. Given $n \in \mathbb{N}$, we denote with $[n]$ the set $\{1, \dots, n\}$, while with $\Pr[A]$ we indicate the probability of the event A . Throughout, q_H is an upper bound on the number of queries the adversary makes to the two random oracles H_{cm} and H_{sig} of Sparkle+, and does not count queries to any random oracle used internally by DS; q_S is an upper bound on the total number of queries it makes to the three signing oracles, counted individually and not by session; and $q = q_H + q_S + 1$. The quantity q is the capacity used by the source reduction for the vector supplied to the forking algorithm; we import, without independently verifying, its accounting that the simulator consumes at most q coordinates. This should not be read as asserting that an execution of the real protocol creates at

most q random-oracle table entries, which it need not, since a single Sign_3 call evaluates $H_{\text{cm}}(i, R_i)$ for every $i \in \mathcal{S}$. In particular q_S bounds the number of Sign_2 calls, which are the ones on which $\mathcal{B}_1$ relays a query to its own signing oracle.

2.2 Sparkle+

We recall the preliminaries required to understand the Sparkle+ scheme as well as our results and proofs. The description we give is that of the full version [14]; as recalled in the introduction, the auxiliary signature scheme DS used in the second signing round is absent from the proceedings version [19].

Definition 1 (Digital Signature). A digital signature scheme DS is a set of polynomial time algorithms $\text{Setup}()$, $\text{KeyGen}()$, $\text{Sign}()$, $\text{Verify}()$, used to ensure authenticity, such that:

- $\text{Setup}(1^\kappa, (\mathbb{G}, p, g)) \rightarrow \text{par}_{DS}$ outputs the digital signature parameters; the group is supplied so that a scheme may reuse it, and is ignored by schemes that generate their own;
- $\text{KeyGen}(\text{par}_{DS}) \rightarrow (\text{pk}, \text{sk})$ is a probabilistic algorithm that outputs a public and secret key pair for a user;
- $\text{Sign}(\text{msg}, \text{sk}) \rightarrow \sigma$ is a probabilistic algorithm that outputs a valid signature σ for a message msg under secret key sk ;
- $\text{Verify}(\text{msg}, \sigma, \text{pk}) \rightarrow 1/0$ is a deterministic algorithm that outputs 1 (accept) if the signature is valid for the message msg under public key pk , or 0 (reject) otherwise.

The scheme is perfectly correct if $\Pr[\text{Verify}(\text{msg}, \text{Sign}(\text{msg}, \text{sk}), \text{pk}) = 1] = 1$ for every message msg and every key pair (pk, sk) in the range of $\text{KeyGen}(\text{par}_{DS})$. We restrict throughout to perfectly correct signature schemes; Schnorr, the instantiation proposed for DS, is one.

A standard example is the Schnorr signature scheme. Let us recap how it works.

Definition 2 (Schnorr Digital Signature). The Schnorr digital signature SCH is a digital signature that works as follows:

- $\text{Setup}(1^\kappa, (\mathbb{G}, p, g)) \rightarrow \text{par}_{SCH}$: the setup algorithm receives the group parameters and selects a hash function, that is $\text{par}_{SCH} = ((\mathbb{G}, p, g), H)$, with $H : \{0, 1\}^* \rightarrow \mathbb{Z}_p$; the description is syntactic, and when the security of Schnorr is invoked below H is modelled as a random oracle. Taking the group as an input rather than generating it is what allows the auxiliary scheme of Sparkle+ to share the group of the threshold scheme, as discussed at the end of this section.
- $\text{KeyGen}(\text{par}_{SCH}) \rightarrow (\text{pk}, \text{sk}) = (X, (x, X))$: this algorithm samples $x \xleftarrow{\$} \mathbb{Z}_p$, computes $X = g^x$, and sets the secret key to the pair (x, X) , so that the signer also holds its own public key.

- $\text{Sign}(m, \text{sk}) \rightarrow \sigma$: on $\text{sk} = (x, X)$, this probabilistic algorithm computes a signature for a message m in the following way: first, sample $r \xleftarrow{\$} \mathbb{Z}_p$, then compute $R \leftarrow g^r$, $c \leftarrow H(X, m, R)$, $z \leftarrow r + cx \bmod p$. The signature is $\sigma = (R, z)$.
- $\text{Verify}(m, \sigma, \text{pk}) \rightarrow 0/1$: the verification algorithm computes $c \leftarrow H(\text{pk}, m, R)$ and checks if $R \cdot \text{pk}^c = g^z$.

A threshold signature scheme (TSS) allows a set of n potential signers to jointly produce a signature on a message that can be verified using a single public key, provided that at least a threshold of $t + 1$ signers participate. Following [14], the signing protocol of Sparkle+ is always run with a set of exactly $t + 1$ signers, and we instantiate the general notion accordingly, so that $|\mathcal{S}| = t + 1$ throughout. Shamir sharing prevents any t or fewer parties from reconstructing the secret key; that they are also unable to produce a signature is a separate property, captured by the threshold unforgeability game of Section 3. Some TSS use secret sharing schemes in order to generate keys. In the Sparkle+ case, Shamir's secret sharing scheme is involved.

Definition 3 (Shamir's Sharing Scheme). Shamir secret sharing is a $(t + 1)$ -out-of- n secret sharing scheme (with secret $x \in \mathbb{Z}_p$, for a prime $p > n$, so that the evaluation points $1, \dots, n$ are distinct in $\mathbb{Z}_p$ and the differences $j - i$ are invertible), consisting of two algorithms. All arithmetic below is in $\mathbb{Z}_p$.

- $\text{IssueShares}(x, n, t + 1) \rightarrow \{(1, x_1), \dots, (n, x_n)\}$.
On input a secret x , the number of participants n , and the threshold $t + 1$, it performs the following. First, define a polynomial $\varphi(Z) = x + a_1Z + a_2Z^2 + \dots + a_tZ^t$ by sampling t coefficients at random: $a_1, \dots, a_t \xleftarrow{\$} \mathbb{Z}_p$. Then, set each participant's share x_i , $i \in [n]$, to be the evaluation of $\varphi(i)$:

$$x_i \leftarrow x + \sum_{j \in [t]} a_j \cdot i^j.$$

Output $\{(i, x_i)\}_{i \in [n]}$.

- $\text{Recover}(t + 1, \{(i, x_i)\}_{i \in \mathcal{S}}) \rightarrow \perp / x$.
On input the threshold $t + 1$ and a set of shares $\{(i, x_i)\}_{i \in \mathcal{S}}$, it outputs $\perp$ if $\mathcal{S} \not\subseteq [n]$ or if $|\mathcal{S}| < t + 1$; reconstruction is thus defined for any $\mathcal{S}$ of size at least $t + 1$, whereas the protocol of Sparkle+ and every game analysed here restrict signing sets to exactly $t + 1$ parties, and we make no claim about larger ones. Otherwise it returns the constant term of the polynomial interpolating the given points, which for honestly generated shares is the original secret x :

$$x \leftarrow \sum_{i \in \mathcal{S}} x_i \lambda_i,$$

where λ_i is defined as:

$$\lambda_i = \prod_{j \in \mathcal{S}, j \neq i} \frac{j}{j - i}.$$

Recovery rests on Lagrange interpolation; privacy is information-theoretic: for any two secrets and any fixed set of at most t indices, the corresponding shares are identically distributed.

Definition 4 (Threshold Signature Scheme [21]). Consider a signature scheme $S=(\text{Setup}, \text{KeyGen}, \text{Sign}, \text{Verify})$. A $(t+1)$ -out-of- n threshold signature scheme *TSS* for S distributes the signing among a group of n players, $P_1, \dots, P_n$, in such a way that any group of at least $t+1$ of them can jointly generate a signature. What groups of at most t players can or cannot do is a security property and not part of the syntax; it is the subject of the unforgeability game of Section 3. Setup and verification are inherited from the underlying scheme S . More formally, a *TSS* adds two algorithms:

- **Thresh-KeyGen**, the key generation algorithm, which takes as input the public parameters and the pair $(n, t+1)$ and returns the public key pk together with a private output sk_i for each P_i , its share of the private key; each sk_i is the private local state of P_i . We do not require of a threshold scheme in general that this state be a share of some pre-existing ordinary signing key: in *Sparkle+* it happens to be $sk_i = (x_i, \hat{sk}_i)$, where the x_i are a $(t+1)$ -out-of- n Shamir sharing of the Schnorr exponent and the auxiliary key $\hat{sk}_i$ is generated independently and is a share of nothing. In *Sparkle+* this algorithm is called simply **KeyGen**. We model it as a trusted dealer: a centralised algorithm run once, before the adversary interacts with the signing oracles, as in the games of Section 3. A distributed key generation protocol could be used instead, as [14] observes, but its execution is not modelled here and none of the bounds below accounts for it.
- **Thresh-Sign**, the distributed signing protocol which takes as public input a message m to be signed as well as a private input sk_i from each involved player. It outputs a signature σ .

No distributed verification algorithm is needed: *Sparkle+* outputs an ordinary Schnorr signature, so that verification is the standard Schnorr verification and requires no interaction among the signers.

The scheme introduced in [14], called *Sparkle+*, is a three-round $(t+1)$ -out-of- n threshold signature scheme, a variant of the Schnorr digital signature. All the tuples given to H_{cm} , to H_{sig} and to $DS.\text{Sign}$ are assumed to be encoded canonically and injectively, with $\mathcal{S}$ written in increasing order and the commitments as the vector indexed by that order; without this the equality of two transcripts would not be well defined.

Definition 5 (Sparkle+ protocol). *Sparkle+* is a $(t+1)$ -out-of- n threshold signature scheme composed of the following polynomial-time algorithms $\text{Setup}()$, $\text{KeyGen}()$, $\text{Sign}_1()$, $\text{Sign}_2()$, $\text{Sign}_3()$, $\text{Combine}()$, $\text{Verify}()$:

- $\text{Setup}(1^\kappa) \rightarrow \text{par}_{TSS}$. Takes as input a string 1^κ based on the security parameter κ . It runs $(\mathbb{G}, p, g) \leftarrow \text{GrGen}(1^\kappa)$, selects two hash functions $H_{cm}, H_{sig} : \{0, 1\}^* \rightarrow \mathbb{Z}_p$, and runs $\text{par}_{DS} \leftarrow DS.\text{Setup}(1^\kappa, (\mathbb{G}, p, g))$, passing its own group to the auxiliary scheme. It outputs $\text{par}_{TSS} = ((\mathbb{G}, p, g), \text{par}_{DS}, H_{cm}, H_{sig})$.

- $\text{KeyGen}(\text{par}_{\text{TSS}}, n, t+1) \rightarrow (\text{pk}, \{(\text{pk}_i, \text{sk}_i)\}_{i \in [n]})$. It takes as input the public parameters, the number of users n and the threshold $t+1$; here and below the public parameters, together with pk and $\{\text{pk}_i\}_{i \in [n]}$ once generated, are available to every algorithm and are not repeated in the signatures of the signing rounds. It samples the master secret $x \xleftarrow{\$} \mathbb{Z}_p$ and sets the joint public key $\text{pk} \leftarrow g^x$. It shares x by $\{(i, x_i)\}_{i \in [n]} \leftarrow \text{IssueShares}(x, n, t+1)$ as in Definition 3, and sets $X_i \leftarrow g^{x_i}$. The keys X_i are not used by the signing rounds; they belong to the key syntax of [14], where they permit a misbehaving signer's share to be identified, and they are used by the simulators of Section 5 and Section 6. It then runs $(\hat{\text{pk}}_i, \hat{\text{sk}}_i) \xleftarrow{\$} \text{DS.KeyGen}(\text{par}_{\text{DS}})$ for every $i \in [n]$, and sets $\text{pk}_i \leftarrow (X_i, \hat{\text{pk}}_i)$ and $\text{sk}_i \leftarrow (x_i, \hat{\text{sk}}_i)$. The output is $(\text{pk}, \{(\text{pk}_i, \text{sk}_i)\}_{i \in [n]})$: the master secret x is not part of it, and is discarded.
- $\text{Sign}_1(k, \text{sk}_k) \rightarrow (cm_k, st_{k,1})$. The first signature round takes as input the user k and its secret key sk_k . A scalar nonce $r_k \xleftarrow{\$} \mathbb{Z}_p$ is sampled randomly, from which the group nonce $R_k \leftarrow g^{r_k}$ is computed, as well as the commitment $cm_k \leftarrow H_{cm}(k, R_k)$, which is returned by the algorithm together with the state

$$st_{k,1} \leftarrow (cm_k, R_k, r_k, \text{sk}_k).$$

The state carries the long-term secret key $\text{sk}_k = (x_k, \hat{\text{sk}}_k)$, which is how the two later rounds obtain it; the public parameters and the public keys pk and $\{\text{pk}_i\}_{i \in [n]}$ are global and available to all algorithms.

- $\text{Sign}_2(k, st_{k,1}, \mathcal{S}, m, \{cm_i\}_{i \in \mathcal{S}}) \rightarrow ((R_k, \hat{\sigma}_k), st_{k,2})$. The second signature round takes as input a user k (part of the set $\mathcal{S}$), its state after the first round of signature $st_{k,1}$, a set of signers $\mathcal{S}$, a message to be signed m and the commitments of all the signers $\{cm_i\}_{i \in \mathcal{S}}$. The algorithm returns $\perp$ if $\mathcal{S} \not\subseteq [n]$, if $|\mathcal{S}| \neq t+1$, if $k \notin \mathcal{S}$, or if the commitment cm_k received among $\{cm_i\}_{i \in \mathcal{S}}$ differs from the one recorded in $st_{k,1}$. Otherwise the message msg_k is built in the following way: $\text{msg}_k \leftarrow (k, cm_k, R_k, \mathcal{S}, m, \{cm_i\}_{i \in \mathcal{S}})$. A signature $\hat{\sigma}_k$ is produced as $\hat{\sigma}_k \leftarrow \text{DS.Sign}(\text{msg}_k, \hat{\text{sk}}_k)$. The group nonce R_k and the key $\hat{\text{sk}}_k$ are parsed from $st_{k,1}$, and the algorithm outputs $(R_k, \hat{\sigma}_k)$ together with the state

$$st_{k,2} \leftarrow (R_k, r_k, cm_k, \hat{\sigma}_k, \mathcal{S}, m, \{cm_i\}_{i \in \mathcal{S}}, \text{sk}_k).$$

- $\text{Sign}_3(k, st_{k,2}, M_2) \rightarrow z_k$, where M_2 is a finite map sending an index to a pair $(R_i, \hat{\sigma}_i)$. The third and last signature round takes as input a user k , its state after the second round of signature $st_{k,2}$ and the collection M_2 of the second-round outputs. It parses $(R_k, r_k, cm_k, \hat{\sigma}_k, \mathcal{S}, m, \{cm_i\}_{i \in \mathcal{S}}, \text{sk}_k)$ from $st_{k,2}$, so that the message, the signer set, the first-round commitments and the secret share x_k are all taken from the state rather than from the round-three input. It first checks that $\text{dom}(M_2) = \mathcal{S}$ and that the pair $(R_k, \hat{\sigma}_k)$ received on behalf of k is the one recorded in $st_{k,2}$, returning $\perp$ if not; we write $(R_i, \hat{\sigma}_i)$ for $M_2(i)$ below. It then verifies, for every $i \in \mathcal{S}$, that the nonce revealed matches the commitment of the first round, $cm_i = H_{cm}(i, R_i)$, and that the auxiliary signature verifies, $\text{DS.Verify}(\text{msg}_i, \hat{\sigma}_i, \hat{\text{pk}}_i) = 1$ with $\text{msg}_i \leftarrow (i, cm_i, R_i, \mathcal{S}, m, \{cm_j\}_{j \in \mathcal{S}})$, returning $\perp$ if either check fails for some

- i. The first check validates each commitment opening; the second, the auxiliary-signature check, verifies every supplied signature against the canonically encoded transcript built from the signer's own stored $(\mathcal{S}, m, \{cm_j\}_{j \in \mathcal{S}})$, which enforces that all accepted auxiliary signatures verify against the same locally stored transcript. What unforgeability of DS gives is that, for an honest party i , an accepted signature is on a message that i 's second round has signed at some point, or else yields an EUF-CMA forgery against DS; that it was produced in the present execution is a stronger statement, requiring an argument about transcript uniqueness which we do not make here. For a corrupted party the check establishes only that whoever holds that party's auxiliary signing key authenticated that transcript. If they pass, the algorithm computes $R \leftarrow \prod_{i \in \mathcal{S}} R_i$, $c \leftarrow H_{sig}(pk, m, R)$ and $z_k \leftarrow r_k + c\lambda_k x_k$, where λ_k is defined as in Definition 3.*
- ***Combine** $(\mathcal{S}, m, \{cm_i, (R_i, \hat{\sigma}_i), z_i\}_{i \in \mathcal{S}}) \rightarrow \sigma$. The combine algorithm takes as input the set of signers $\mathcal{S}$, the message to be signed m and all the information created by the signers in the rounds of signature, of which it uses only the pairs (R_i, z_i) : it is a pure aggregation step: in an honest execution it is invoked on outputs already validated by the signing rounds, and the algorithm itself performs no verification. It therefore offers no robustness, a malicious signer being able to submit an arbitrary z_i and cause the combined signature to fail; that is a liveness matter and not an unforgeability one. We keep the syntax of [14], in which the unused arguments are present, rather than reducing it to $\mathcal{S}$ and $\{(R_i, z_i)\}_{i \in \mathcal{S}}$. It recovers $R \leftarrow \prod_{i \in \mathcal{S}} R_i$, computes $z \leftarrow \sum_{i \in \mathcal{S}} z_i \bmod p$, and returns $\sigma \leftarrow (R, z)$.*
 - ***Verify** $(m, \sigma, pk) \rightarrow 0/1$. This verification algorithm takes as input the public key of the system pk , the message m and its signature $\sigma = (R, z)$. It computes $c \leftarrow H_{sig}(pk, m, R)$ and checks if $R \cdot pk^c \stackrel{?}{=} g^z$ and outputs 1 if the check is correct, 0 otherwise.*

Correctness is immediate, given the assumed correctness of DS, which makes the verification of the auxiliary signatures in **Sign₃** succeed on honest inputs. If all signers follow the protocol then, writing $r = \sum_{i \in \mathcal{S}} r_i \bmod p$, so that $R = \prod_{i \in \mathcal{S}} g^{r_i} = g^r$, and $c = H_{sig}(pk, m, R)$,

$$z = \sum_{k \in \mathcal{S}} z_k = \sum_{k \in \mathcal{S}} r_k + c \sum_{k \in \mathcal{S}} \lambda_k x_k = r + cx,$$

since $|\mathcal{S}| = t + 1$ and the Lagrange coefficients for $\mathcal{S}$ reconstruct x from the shares $\{x_k\}_{k \in \mathcal{S}}$; hence $g^z = R \cdot pk^c$ and the signature verifies.

When DS is instantiated with a Schnorr signature, as [14] proposes and as Section 7.3 assumes, we take it to use the same group $(\mathbb{G}, p, g)$, the setup of **Sparkle+** passing its own group to **DS.Setup**, and a random oracle independent of H_{cm} and H_{sig} : all three are obtained from a single master random oracle restricted to disjoint tagged domains, which makes the independence exact in the model. Sharing the group is what allows Section 7.3 to express the advantage of DS in terms of the same p . It does not by itself justify using the same resource

parameter: the queries that the adversary makes to the oracle of DS are counted by a separate budget q_{DS} , and the identification $q_{\text{DS}} \leq q$ is one of the modelling conventions declared in Section 7.3, not a consequence of the syntax. The two hash functions have codomain $\mathbb{Z}_p$. This will matter in Section 4, where the set H of the forking lemma is instantiated with $\mathbb{Z}_p$ and $|H| = p$.

3 Security Assumptions

We recap the cryptographic assumptions on which the security of Sparkle+ is based. Let us start distinguishing between the static and the adaptive case. Throughout, the two hash functions H_{cm} and H_{sig} of Sparkle+ are modelled as independent random oracles, in the *Random Oracle Model* (ROM); this provides the interface that the reductions simulate and program, subject to the consistency arguments and bad-event bounds of the source proofs. Static security relies in addition on the *Discrete Logarithm* (DL) assumption and on the *Existential Unforgeability* (EUF-CMA) of the auxiliary digital signature scheme DS, the latter being a property of DS alone and not itself a ROM notion; if DS is instantiated with a Schnorr signature, its own proof of unforgeability also uses the ROM. Adaptive security up to $\lfloor t/2 \rfloor$ corruptions replaces DL by one of its variants, the *Algebraic One-More Discrete Logarithm* (AOMDL) assumption.

Definition 6. *In the Random Oracle Model a hash function $H: \{0,1\}^* \rightarrow \mathcal{Y}$ is replaced by an oracle that answers a fresh query with an element drawn uniformly from $\mathcal{Y}$ and repeats its earlier answer on a repeated query, the same oracle being available to the adversary, to the scheme and to any reduction. Distinct hash functions are replaced by independent such oracles.*

The discrete logarithm assumption is formulated through the game of Table 1, and its algebraic one-more variant through that of Table 2. Let $\mathcal{A}$ be a *probabilistic polynomial time* (PPT) adversary, let κ be a security parameter and let $\text{GrGen}(1^\kappa)$ be a randomised algorithm returning a group description $(\mathbb{G}, p, g)$ with p prime and $\log p = \Theta(\kappa)$.

MAIN Game $_{\mathcal{A}}^{\text{DL}}(\kappa)$
$(\mathbb{G}, p, g) \leftarrow \text{GrGen}(1^\kappa)$ $x \xleftarrow{\$} \mathbb{Z}_p; X \leftarrow g^x$ $x' \xleftarrow{\$} \mathcal{A}((\mathbb{G}, p, g), X), x' \in \mathbb{Z}_p$ if $x' = x$ return 1 return 0

Table 1. The DL game: an adversary $\mathcal{A}$ receives group parameters $(\mathbb{G}, p, g)$ and a challenge $X \in \mathbb{G}$. $\mathcal{A}$, in order to win, wants to output x such that $g^x = X$.

Definition 7. We define the advantage of an adversary $\mathcal{A}$ for the discrete logarithm game described in Table 1 as follows:

$$\mathbf{Adv}_{\mathcal{A}}^{\text{DL}}(\kappa) = \Pr[\text{Game}_{\mathcal{A}}^{\text{DL}}(\kappa) = 1].$$

The discrete logarithm assumption holds if $\mathbf{Adv}_{\mathcal{A}}^{\text{DL}}(\kappa)$ is negligible for every PPT adversary.

For what concerns the AOMDL assumption, we need a slightly more complex game, described in Table 2.

MAIN $\text{Game}_{\mathcal{A}}^{t+1\text{-AOMDL}}(\kappa, t)$	$\mathcal{O}^{\text{DL}}(X, \alpha, \{\beta_i\}_{i=0}^t)$
$(\mathbb{G}, p, g) \leftarrow \text{GrGen}(1^\kappa)$ $\text{ctr} \leftarrow 0$ for $i \in [0..t]$ do $x_i \xleftarrow{\$} \mathbb{Z}_p; X_i \leftarrow g^{x_i}$ $\vec{x} \leftarrow (x_0, \dots, x_t)$ $\vec{X} \leftarrow (X_0, X_1, \dots, X_t)$ $\vec{x}' \leftarrow \mathcal{A}^{\mathcal{O}^{\text{DL}}}((\mathbb{G}, p, g), \vec{X})$ if $\vec{x}' = \vec{x}$ $\quad \textbf{return } 1$ return 0	$// \alpha, \beta_0, \dots, \beta_t \in \mathbb{Z}_p$ return $\perp$ if $\text{ctr} = t$ return $\perp$ if $X \neq g^\alpha \prod_{i=0}^t X_i^{\beta_i}$ $\text{ctr} \leftarrow \text{ctr} + 1$ $x \leftarrow \alpha + \sum_{i=0}^t x_i \beta_i$ return x

Table 2. The AOMDL game: an adversary $\mathcal{A}$ receives a vector of $t+1$ discrete logarithm challenges $(X_0, \dots, X_t)$. $\mathcal{A}$ is equipped with a discrete logarithm oracle $\mathcal{O}^{\text{DL}}$. The latter answers at most t valid queries, malformed representation queries being rejected, and only on elements $X \in \mathbb{G}$ for which $\mathcal{A}$ supplies a representation $(\alpha, \{\beta_i\}_{i=0}^t) \in \mathbb{Z}_p^{t+2}$ with $X = g^\alpha \prod_{i=0}^t X_i^{\beta_i}$; the oracle rejects the query if the representation does not check out, so that the value it returns is always $\log_g X$. The game is parametrised by t , supplied as an input before the group is generated and the challenges sampled. The reductions of Section 6 therefore require t to be fixed before the experiment begins, whereas the game of Table 4 has the threshold adversary output n and $t+1$ itself. Accordingly n and $t+1$ are inputs to the game of Table 4, where [14] has the adversary output them; we depart from the source on this point, since otherwise a reduction could not know which $t+1$ -AOMDL experiment to enter. At the end of the game, the adversary needs to output all the $t+1$ discrete logarithms given as challenge.

Definition 8. We define the advantage of an adversary $\mathcal{A}$ for the algebraic one more discrete logarithm game described in Table 2 as follows:

$$\mathbf{Adv}_{\mathcal{A}}^{t+1\text{-AOMDL}}(\kappa, t) = \Pr[\text{Game}_{\mathcal{A}}^{t+1\text{-AOMDL}}(\kappa, t) = 1].$$

The algebraic one more discrete logarithm assumption holds if, for every PPT adversary $\mathcal{A}$ and every polynomially bounded $t = t(\kappa)$, the parameter function being fixed before the experiment is sampled rather than chosen adaptively by $\mathcal{A}$, the

quantity $\text{Adv}_{\mathcal{A}}^{t+1\text{-AOMDL}}(\kappa, t)$ is negligible. The coefficients $\alpha, \beta_0, \dots, \beta_t$ supplied to the oracle and the values returned by $\mathcal{A}$ are elements of $\mathbb{Z}_p$. We reproduce the game of [14, Fig. 5]; in the reductions of Section 6 we abbreviate $\text{Adv}_{\mathcal{B}_2'}^{t+1\text{-AOMDL}}(\kappa, t)$ to $\text{Adv}_{\mathcal{B}_2'}^{t+1\text{-AOMDL}}(\kappa)$, the value of t being fixed by the surrounding game.

As we said before, we need the digital signature chosen for the Sparkle+ scheme to satisfy EUF-CMA security. Correctness of DS, namely that

$$\text{Verify}(\text{msg}, \text{Sign}(\text{msg}, \text{sk}), \text{pk}) = 1$$

for every message and every key pair produced by **KeyGen**, is assumed throughout. The corresponding security game is described in Table 3; correctness was defined above.

MAIN Game $_{\mathcal{A}, \text{DS}}^{\text{EUF-CMA}}(\kappa)$	$\mathcal{O}^{\text{DS.Sig}}(\text{msg})$
$(\mathbb{G}, p, g) \leftarrow \text{GrGen}(1^\kappa)$	$Q_{\text{msg}} \leftarrow Q_{\text{msg}} \cup \{\text{msg}\}$
$\text{par}_{\text{DS}} \leftarrow \text{DS.Setup}(1^\kappa, (\mathbb{G}, p, g))$	$\hat{\sigma} \leftarrow \text{DS.Sign}(\text{msg}, \hat{\text{sk}})$
$Q_{\text{msg}} \leftarrow \emptyset$	return $\hat{\sigma}$
$(\hat{\text{pk}}, \hat{\text{sk}}) \xleftarrow{\$} \text{DS.KeyGen}(\text{par}_{\text{DS}})$	
$(\text{msg}^*, \hat{\sigma}^*) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}^{\text{DS.Sig}}}(\text{par}_{\text{DS}}, \hat{\text{pk}})$	
return 0 if $\text{msg}^* \in Q_{\text{msg}}$ $\vee \text{DS.Verify}(\text{msg}^*, \hat{\sigma}^*, \text{pk}) \neq 1$	
return 1	

Table 3. The EUF-CMA security game for a digital signature scheme DS. The group is generated inside the game and passed to **DS.Setup**, which ignores it for schemes that do not reuse it; in the corresponding reduction the group generated by the EUF-CMA challenger is adopted as the Sparkle+ group, no second group being generated. Challenger and scheme use the same **GrGen**, the fixed generator associated with the Sparkle+ parameters, and unforgeability of DS is understood relative to that setup distribution whenever DS reuses the group.

Definition 9. Given a digital signature DS, we define the advantage of an adversary $\mathcal{A}$ for the EUF-CMA game described in Table 3 as

$$\text{Adv}_{\mathcal{A}, \text{DS}}^{\text{EUF-CMA}}(\kappa) = \Pr[\text{Game}_{\mathcal{A}, \text{DS}}^{\text{EUF-CMA}}(\kappa) = 1].$$

A digital signature DS is existentially unforgeable under chosen message attacks if for all PPT adversaries $\mathcal{A}$, $\text{Adv}_{\mathcal{A}, \text{DS}}^{\text{EUF-CMA}}(\kappa)$ is negligible.

3.1 Threshold Schemes Security Properties

Let $\text{frac} \in \{1/2, 1\}$ be the fraction that fixes the maximum number of users an adversary may corrupt; these are the only two values used in [14] and here. Security of threshold schemes is discussed against two different adversarial models:

- **static adversary.** This kind of model is able to corrupt up to $\lfloor \text{frac} \cdot t \rfloor$ parties among the n users. A static adversary must declare those parties before key generation takes place, and in particular before knowing which parties will effectively take part in any signature.
- **adaptive adversary.** These adversaries can also corrupt $\lfloor \text{frac} \cdot t \rfloor$ parties, but in a more realistic way. In fact, an adaptive adversary may name an initial corrupt set before key generation, exactly as a static one does, and may afterwards issue corruption requests at arbitrary times through $\mathcal{O}^{\text{Corrupt}}$, choosing the party as a function of the transcript observed so far, and receives that party’s secret key together with its state in every signing session opened up to that moment.

[14] proves **Sparkle+** secure against static adversaries with $\text{frac} = 1$ and against adaptive adversaries with $\text{frac} = 1/2$, and a theorem is stated for adaptive adversaries with $\text{frac} = 1$, whose proof was subsequently called into question by the works cited in the introduction. In this paper we analyse only the first two cases.

In order to define the security for **Sparkle+**, we must define proper games. The existential unforgeability game described in Table 4 is required for a three-round threshold signature scheme. Five conventions govern that game. All algorithms and all adversaries have access to the two random oracles $\mathcal{H}_{\text{cm}}$ and $\mathcal{H}_{\text{sig}}$, which we leave implicit in the table; the adversary’s direct queries to them are the ones bounded by q_H , accesses made internally by the scheme or the simulator being accounted for separately in the source reduction. The keys are those produced by **KeyGen** in Section 2, so that $\text{pk}_i = (X_i, \hat{\text{pk}}_i)$ and $\text{sk}_i = (x_i, \hat{\text{sk}}_i)$, both components of a corrupted party’s key being handed to the adversary. Entries of Q_{st} that have never been assigned are $\perp$, and entries once assigned are never erased, so that a corruption returns the state of every session opened by that party up to that point; **Sparkle+** is analysed without assuming secure erasures; in particular the round-one and round-two entries are retained when round three completes, so a later corruption exposes the ephemeral nonce information they contain. Execution identifiers are fresh by construction, $\mathcal{O}^{\text{Sign}_1}$ drawing each one from outside the set Q_{eid} of those already issued. A message is added to Q_m as soon as it is submitted to $\mathcal{O}^{\text{Sign}_2}$, before **Sign**₂ is called, so that m ceases to qualify as a forgery even when that call is rejected; the set therefore records the messages submitted to an honest party’s second-round interface and not the messages on which a threshold signature was completed. This is the freshness notion of [14], which is more restrictive than the latter, and we analyse it as it stands since, with respect to the final forgery condition, it can only restrict the adversary.

Three further points of typing. Execution identifiers are local: each honest party holds its own handle eid_i , returned by its own $\mathcal{O}^{\text{Sign}_1}$ call, and the eid argument of the later oracles is the handle of the party being called, the protocol messages of the others being supplied as a collection indexed by $\mathcal{S}$; the subscript eid_i identifies the session each message came from and is not itself an input to **Sign**₃. In $\mathcal{O}^{\text{Sign}_3}$ the set $\mathcal{S}$ is the index domain of that collection, and is compared with the one recorded in $\text{st}_{k,\text{eid},2}$. Finally, the two random oracles are independent,

answered by lazy sampling from $\mathbb{Z}_p$ and consistent on repeated queries, and are shared by the adversary, the scheme and the reductions; the game of Table 3 is the generic EUF-CMA syntax, and when DS is instantiated in the random oracle model its own oracle is added to that game and is not counted by q_H . In order to not be redundant in the definitions, we will use the $\boxed{\text{dashbox}}$ to refer to algorithms that are called only for the adaptive case. If they are not called, we obtain the static security definitions instead.

Definition 10. *Let us define the advantage of an adversary $\mathcal{A}$ against the game described in Table 4 as:*

$$\boxed{\text{adp}}_{\mathcal{A}, \overline{\text{TS}}}^{\text{TS-EUF-CMA}}(\kappa, n, t+1, \text{frac}) = \Pr[\text{Game}_{\mathcal{A}, \overline{\text{TS}}}^{\text{TS-EUF-CMA}}(\kappa, n, t+1, \text{frac}) = 1].$$

Where n and t are fixed by the surrounding statement we suppress them and write $\boxed{\text{adp}}_{\mathcal{A}, \overline{\text{TS}}}^{\text{TS-EUF-CMA}}(\kappa, \text{frac})$, as we do for the AOMDL advantage above. We say that a threshold scheme is existentially unforgeable under chosen message attacks against a static/ $\boxed{\text{adaptive}}$ adversary if, for every PPT adversary $\mathcal{A}$ and all polynomially bounded $n = n(\kappa)$ and $t = t(\kappa)$ with $1 \leq t < n < p$, the quantity $\boxed{\text{adp}}_{\mathcal{A}, \overline{\text{TS}}}^{\text{TS-EUF-CMA}}(\kappa, n, t+1, \text{frac})$ is negligible.

4 The Forking Step

Both proofs we revisit rewind the simulator in order to obtain the additional algebraic information needed to solve their underlying problem, a discrete logarithm in the static case and a $t+1$ -AOMDL instance in the adaptive one, and both invoke the general forking lemma of Bellare and Neven [7]. The lemma admits two formulations: a lower bound on the acceptance probability of the forking algorithm in terms of the acceptance probability of the simulator, and an upper bound on the latter in terms of the former. The second follows from the first, and it is what the proof requires. Since the discrepancies described in Section 5 and Section 6 depend on which of the two is used, we record both formulations together with the exact inversion that links them.

For this section, let ϵ denote the empty string.

Fix an integer $q \geq 1$ and a finite set H with $|H| \geq 2$; let IG be a randomised input generator and let $\mathcal{R}$ be a non-empty set of random tapes, a tape ρ carrying all the internal randomness of the algorithm below, so that its execution is determined once x , $(h_1, \dots, h_q)$ and ρ are fixed. Let SIM be a probabilistic algorithm which, on input x in the range of IG , a tuple $(h_1, \dots, h_q) \in H^q$ and a tape $\rho \in \mathcal{R}$, outputs a pair

$$(I, \sigma) = \text{SIM}(x, h_1, \dots, h_q; \rho), \quad I \in \{0, \dots, q\}, \sigma \in \{0, 1\}^*,$$

the output being determined by those three inputs. The index I designates the coordinate at which the fork is to be made, with $I = 0$ signalling failure; no

$\text{MAIN Game}_{\mathcal{A}, \text{TS}}^{\text{adp}, \text{TS-EUF-CMA}}(\kappa, n, t+1, \text{frac} \cdot t, \mathcal{O}^{\text{Sign}_1}(k))$ $\text{par} \leftarrow \text{Setup}(1^\kappa)$ return $\perp$ if $\neg(n, t \in \mathbb{N} \wedge 1 \leq t < n < p)$ $Q_{\text{st}} \leftarrow \emptyset, Q_m \leftarrow \emptyset, Q_{\text{eid}} \leftarrow \emptyset$ $(\text{cor}, \text{st}_{\mathcal{A}}) \xleftarrow{\\$} \mathcal{A}(\text{par}, n, t+1)$ return $\perp$ if $\text{cor} \not\subseteq [n]$ $\vee \text{cor} > \lfloor \text{frac} \cdot t \rfloor$ $\text{hon} \leftarrow [n] \setminus \text{cor}$ $(\text{pk}, \{(\text{pk}_i, \text{sk}_i)\}_{i \in [n]}) \xleftarrow{\\$} \text{KeyGen}(\text{par}, n, t + \mathcal{O}^{\text{Sign}_2}(k, \text{eid}, \mathcal{S}, m, \{\text{pm}_{i, \text{eid}_i, 1}\}_{i \in \mathcal{S}}))$ $\text{in} \leftarrow (\text{pk}, \{\text{pk}_i\}_{i \in [n]}, \{\text{sk}_j\}_{j \in \text{cor}}, \text{st}_{\mathcal{A}})$ $(m^*, \sigma^*) \xleftarrow{\\$} \mathcal{A}^{\mathcal{O}^{\text{Sign}_1, \text{Sign}_2, \text{Sign}_3}, \text{Corrupt}}}(\text{in})$ return 1 if $m^* \notin Q_m$ $\wedge \text{Verify}(m^*, \sigma^*, \text{pk}) = 1$ return 0 $\mathcal{O}^{\text{Corrupt}}(k)$ return $\perp$ if $k \notin \text{hon} \vee \text{cor} \geq \lfloor \text{frac} \cdot t \rfloor$ $\text{cor} \leftarrow \text{cor} \cup \{k\}$ $\text{hon} \leftarrow \text{hon} \setminus \{k\}$ $\text{st}_k \leftarrow Q_{\text{st}}[k, \cdot, \cdot]$ //all state for party k return $(\text{sk}_k, \text{st}_k)$	$\mathcal{O}^{\text{Sign}_1}(k)$ return $\perp$ if $k \notin \text{hon}$ $\text{eid} \xleftarrow{\\$} \{0, 1\}^\kappa \setminus Q_{\text{eid}}$ $Q_{\text{eid}} \leftarrow Q_{\text{eid}} \cup \{\text{eid}\}$ $(\text{pm}_{k, \text{eid}, 1}, \text{st}_{k, \text{eid}, 1}) \leftarrow \text{Sign}_1(k, \text{sk}_k)$ $Q_{\text{st}}[k, \text{eid}, 1] \leftarrow \text{st}_{k, \text{eid}, 1}$ return $(\text{eid}, \text{pm}_{k, \text{eid}, 1})$ $\mathcal{O}^{\text{Sign}_2}(k, \text{eid}, \mathcal{S}, m, \{\text{pm}_{i, \text{eid}_i, 1}\}_{i \in \mathcal{S}})$ return $\perp$ if $k \notin \text{hon}$ return $\perp$ if $Q_{\text{st}}[k, \text{eid}, 1] = \perp$ return $\perp$ if $Q_{\text{st}}[k, \text{eid}, 2] \neq \perp$ $\text{parse } \text{st}_{k, \text{eid}, 1} \leftarrow Q_{\text{st}}[k, \text{eid}, 1]$ $Q_m \leftarrow Q_m \cup \{m\}$ $(\text{pm}_{k, \text{eid}, 2}, \text{st}_{k, \text{eid}, 2}) \leftarrow \text{Sign}_2(k, \text{st}_{k, \text{eid}, 1}, \mathcal{S}, m, \{\text{pm}_{i, \text{eid}_i, 1}\}_{i \in \mathcal{S}})$ return $\perp$ if $\text{pm}_{k, \text{eid}, 2} = \perp$ $Q_{\text{st}}[k, \text{eid}, 2] \leftarrow \text{st}_{k, \text{eid}, 2}$ return $\text{pm}_{k, \text{eid}, 2}$ $\mathcal{O}^{\text{Sign}_3}(k, \text{eid}, M_2), M_2 = \{i \mapsto \text{pm}_{i, \text{eid}_i, 2}\}$ return $\perp$ if $k \notin \text{hon}$ return $\perp$ if $Q_{\text{st}}[k, \text{eid}, 2] = \perp$ return $\perp$ if $Q_{\text{st}}[k, \text{eid}, 3] \neq \perp$ $\text{parse } \text{st}_{k, \text{eid}, 2} \leftarrow Q_{\text{st}}[k, \text{eid}, 2]$ $\text{pm}_{k, \text{eid}, 3} \leftarrow \text{Sign}_3(k, \text{st}_{k, \text{eid}, 2}, M_2)$ return $\perp$ if $\text{pm}_{k, \text{eid}, 3} = \perp$ $Q_{\text{st}}[k, \text{eid}, 3] \leftarrow \text{pm}_{k, \text{eid}, 3}$ return $\text{pm}_{k, \text{eid}, 3}$
--	--

Table 4. This game is a variant of an existential unforgeability game in the setting of a three round threshold signature. The aim of the adversary $\mathcal{A}$ is to generate a valid message/signature pair (m^*, σ^*) for a message m^* never submitted to $\mathcal{O}^{\text{Sign}_2}$. The game follows [14, Fig. 6], except that n and $t+1$ are supplied externally rather than output by the adversary, for the reason given in the caption of Table 2. With the quantification over parameter functions used in the definition below, this is the fixed-parameter form of the source’s notion; we do not claim the two equivalent, and every statement in this paper is made for the fixed-parameter form. $\mathcal{A}$ is equipped with three signing oracles and, in the adaptive case, with a corruption oracle too. The oracles test only the state of the game; the checks on the signing set and on the consistency of the protocol messages with the stored state, namely that $\mathcal{S} \subseteq [n]$, that $|\mathcal{S}| = t+1$, that $k \in \mathcal{S}$, and that the commitments and nonces supplied in the later rounds are those recorded in the state of the earlier ones, are performed inside Sign_2 and Sign_3 themselves (Section 2), which return $\perp$ when they fail.

condition is imposed on how σ depends on the coordinates, and in particular it may depend on those after I , which is why the whole suffix is resampled in Table 5.

Definition 11. *The acceptance probability of SIM is*

$$\text{accept}(\text{SIM}) = \Pr \left[I > 0 : \begin{array}{l} x \leftarrow \text{IG}, \quad (h_1, \dots, h_q) \xleftarrow{\$} H^q, \quad \rho \xleftarrow{\$} \mathcal{R}, \\ (I, \sigma) \leftarrow \text{SIM}(x, h_1, \dots, h_q; \rho) \end{array} \right],$$

the three samplings being mutually independent.

$\text{Fork}^{\text{SIM}}(x)$
$\rho \xleftarrow{\$} \mathcal{R}$ $(h_1, \dots, h_q) \xleftarrow{\$} H^q$ $(I, \sigma) \leftarrow \text{SIM}(x, h_1, \dots, h_q; \rho)$ if $I = 0$ return $(0, \epsilon, \epsilon)$ $(h'_1, \dots, h'_q) \xleftarrow{\$} H^{q-I+1}$ $(I', \sigma') \leftarrow \text{SIM}(x, h_1, \dots, h_{I-1}, h'_1, \dots, h'_q; \rho)$ if $(I = I' \text{ and } h_I \neq h'_I)$ return $(1, \sigma, \sigma')$ else return $(0, \epsilon, \epsilon)$

Table 5. The forking algorithm, run on a fixed input x .

Definition 12. *For each fixed x in the range of IG, the forking algorithm $\text{Fork}^{\text{SIM}}(x)$ is defined as in Table 5. Its acceptance probability is taken over the choice of x as well as over the internal randomness:*

$$\text{accept}(\text{Fork}^{\text{SIM}}) = \Pr[b = 1 : x \leftarrow \text{IG}, (b, \sigma, \sigma') \leftarrow \text{Fork}^{\text{SIM}}(x)].$$

Where later sections write $\text{accept}(\text{Fork}^{\text{SIM}}(X))$, with X the instance supplied by the challenger of the underlying game, the quantity meant is this one, the average over the challenger's choice of X ; it is the quantity that the advantage of the reduction is compared with.

Theorem 13 (General Forking Lemma, [7, Lemma 1]). *With the notation and the hypotheses $q \geq 1$, $|H| \geq 2$ used above,*

$$\text{accept}(\text{Fork}^{\text{SIM}}) \geq \text{accept}(\text{SIM}) \left(\frac{\text{accept}(\text{SIM})}{q} - \frac{1}{|H|} \right). \quad (1)$$

The final combination requires an upper bound on $\text{accept}(\text{SIM})$ in terms of $\text{accept}(\text{Fork}^{\text{SIM}})$, obtained by inverting Equation (1). We isolate it, since it is the only step in which the quantity $1/|H|$ intervenes.

Lemma 14 (Inversion of the forking lemma). *Let $\gamma = \text{accept}(\text{SIM})$ and $F = \text{accept}(\text{Fork}^{\text{SIM}})$, and write $N = |H|$. Then Equation (1) implies*

$$\gamma \leq \frac{q}{2N} + \sqrt{\frac{q^2}{4N^2} + qF}. \quad (2)$$

Proof. Equation (1) reads $F \geq \gamma^2/q - \gamma/N$, that is

$$\gamma^2 - \frac{q}{N}\gamma - qF \leq 0.$$

The left-hand side is a quadratic polynomial in γ with positive leading coefficient, hence it is non-positive exactly on the closed interval bounded by its two real roots

$$\frac{1}{2} \left(\frac{q}{N} \pm \sqrt{\frac{q^2}{N^2} + 4qF} \right),$$

and in particular γ does not exceed the larger one, which is the right-hand side of Equation (2). $\square$

For non-negative scalars γ and F , Equation (2) is in fact equivalent to the numerical inequality Equation (1), and not merely implied by it: the smaller root of the quadratic is non-positive, since $\sqrt{q^2/N^2 + 4qF} \geq q/N$, so that a non-negative γ satisfies the quadratic inequality exactly when it is at most the larger root. In the regime where the right-hand side of Equation (2) is at most 1, which is the only one in which the bound says anything, it is therefore the pointwise sharpest scalar upper bound on γ obtainable from Equation (1) alone; outside that regime the trivial bound $\gamma \leq 1$ is better. It is of course not an equality between γ and F . Relaxing Equation (2) once gives the form in which the forking lemma is also recorded in [14].

Corollary 15. *With the notation of Lemma 14,*

$$\text{accept}(\text{SIM}) \leq \frac{q}{|H|} + \sqrt{q \cdot \text{accept}(\text{Fork}^{\text{SIM}})}. \quad (3)$$

Proof. Apply $\sqrt{x+y} \leq \sqrt{x} + \sqrt{y}$, valid for $x, y \geq 0$, to Equation (2) with $x = q^2/4N^2$ and $y = qF$: this gives $\gamma \leq \frac{q}{2N} + \frac{q}{2N} + \sqrt{qF}$. $\square$

Two consequences will be used repeatedly. First, no negligibility argument is needed at this point: the term $1/|H|$ is carried exactly by Equation (2) and, if one prefers a simpler expression, is absorbed into the additive $\frac{q}{|H|}$ of Equation (3). Second, both Equation (2) and Equation (3) leave an additive term *outside* the square root: $\frac{q}{2|H|}$ in the former and $\frac{q}{|H|}$ in the latter.

It remains to say how Sparkle+ instantiates this interface. This section does not audit whether the source simulators satisfy the hypotheses of the general forking lemma: we condition on that applicability and analyse only the numerical consequence of the inequality once it is granted. Four things are imported and not established here, namely that the simulators of [14] are representable in the

form required by Theorem 13 with the resampling semantics used there; that $q = q_H + q_S + 1$ coordinates suffice, an accounting we do not independently verify; that the index returned by the simulator is the one described below; and, in the adaptive case, that the persistent state of the discrete logarithm oracle is observationally irrelevant across the fork. What follows describes the instantiation rather than establishing it.

Both random oracles H_{cm} and H_{sig} have codomain $\mathbb{Z}_p$, so $H = \mathbb{Z}_p$ and $|H| = p$. We regard the two as a single lazy-sampled oracle on the disjoint union of their domains, queries being tagged (cm, u) or (sig, u) ; their fresh answers can then be ordered into one vector in $\mathbb{Z}_p^q$. The tuple $(h_1, \dots, h_q)$ is consumed in order by the simulator of Section 5: each first-time query to either oracle is answered with the next unused coordinate, repeated queries being answered from a table so that the simulation is consistent, and the values that the simulator programs while answering signing queries are taken from the same tuple. When DS has a random oracle of its own, as in the Schnorr instantiation, the randomness defining it is part of ρ and is held fixed across the two executions; the vector $(h_1, \dots, h_q)$ carries only the answers of H_{cm} and H_{sig} , which are the oracles subject to the fork. The quantity $q = q_H + q_S + 1$ is a capacity rather than an exact count: q_H coordinates suffice for the adversary’s own queries, q_S for those programmed during signing calls, since under the source proof’s accounting no such call causes the simulator to program more than one new entry, and one more is reserved for the H_{sig} input on which the forgery is based, in case the adversary never queries it explicitly. A run that uses fewer coordinates simply leaves the remainder unread. The index I returned by the simulator is the coordinate assigned to the H_{sig} query on which the final forgery depends, so that resampling coordinate I in the second run of Table 5 resamples that challenge; the identification is made explicit in Section 5.

The simulator SIM' of Section 6 is read in the same way, with one difference: it also queries the discrete logarithm oracle of the $t + 1$ -AOMDL game, whose state persists across the two runs, so that it is not on the face of it a function of $(x, h_1, \dots, h_q; \rho)$ alone. Treating it as harmless requires three things, which we take from the source rather than verify: that for a fixed challenge instance every valid query determines a unique group element and hence a unique discrete logarithm modulo p , so that repeated queries receive the same answer; that the reduction evaluates that function through its challenger rather than computing it, introducing no further randomness; and that the simulator cannot inspect the counter or any other challenger state. Granted these, the persistent state never changes an answer nor causes a rejection.

5 The Static Bound

The first result of [14] asserts the static security of Sparkle+ under the discrete logarithm assumption and the EUF-CMA security of DS, in the random oracle model. We reproduce the statement.

Theorem 16 ([14, Theorem 1]).

Let GrGen be a group generator and DS be a digital signature scheme. For any PPT adversary $\mathcal{A}$ for the game $\text{Game}_{\mathcal{A}, \text{Sparkle}+}^{\text{TS-EUF-CMA}}(\kappa, \text{frac} = 1)$, there exists a PPT adversary $\mathcal{B}_1$ for the game $\text{Game}_{\mathcal{B}_1, \text{DS}}^{\text{EUF-CMA}}(\kappa)$ making at most q_S queries to its signing oracle, and a PPT adversary $\mathcal{B}_2$ for the game $\text{Game}_{\mathcal{B}_2}^{\text{DL}}(\kappa)$ such that

$$\text{Adv}_{\mathcal{A}, \text{Sparkle}+}^{\text{TS-EUF-CMA}}(\kappa) \leq \sqrt{q \cdot \text{Adv}_{\mathcal{B}_2}^{\text{DL}}(\kappa) + \frac{q^2}{2p} + \frac{2q}{p} + 2n \text{Adv}_{\mathcal{B}_1, \text{DS}}^{\text{EUF-CMA}}(\kappa)}, \quad (4)$$

where n is the number of signers, t is the maximum number of corrupt signers, p is the group size, and $q = q_H + q_S + 1$, where q_H is the number of queries $\mathcal{A}$ can make to the random oracles and q_S is the number of queries $\mathcal{A}$ can make to the signing oracles.

The notation of the statement is that of the source. Under the fixed-parameter convention of Section 3 we read its left-hand side as $\text{Adv}_{\mathcal{A}, \text{Sparkle}+}^{\text{TS-EUF-CMA}}(\kappa, n, t + 1, 1)$, abbreviated to $\text{Adv}_{\mathcal{A}, \text{Sparkle}+}^{\text{TS-EUF-CMA}}(\kappa)$ where n and t are fixed by the context, and we use that reading from here on.

Throughout this section we abbreviate

$$A = q \cdot \text{Adv}_{\mathcal{B}_2}^{\text{DL}}(\kappa), \quad B = \frac{q^2}{2p} + \frac{2q}{p}, \quad C = n \cdot \text{Adv}_{\mathcal{B}_1, \text{DS}}^{\text{EUF-CMA}}(\kappa), \quad (5)$$

so that the right-hand side of Equation (4) is $\sqrt{A + B + 2C}$.

5.1 The relations supplied by the source proof

The proof of Theorem 16 given in [14] proceeds by a sequence of three games. GAME 0 is the TS-EUF-CMA game of Table 4 played against $\text{Sparkle}+$; GAME 1 rules out the event that $\mathcal{A}$ forges a DS signature on the protocol view of an honest party; GAME 2 rules out three further bad events and is the game simulated by the algorithm SIM , which plays the role of the algorithm SIM of Section 4. Writing W_i for the event that $\mathcal{A}$ wins GAME i , the proof establishes the following four relations, whose numbering in the June 2025 revision of [14], that is v4 of Table 8, we recall in brackets (with some simplifications to make (9) more readable).

$$\text{Adv}_{\mathcal{A}, \text{Sparkle}+}^{\text{TS-EUF-CMA}}(\kappa, \text{frac} = 1) = \Pr[W_0] \quad [\text{their (8)}] \quad (6)$$

$$|\Pr[W_0] - \Pr[W_1]| \leq C \quad [\text{their (10)}] \quad (7)$$

$$|\Pr[W_1] - \Pr[W_2]| \leq B \quad [\text{their (11)}] \quad (8)$$

$$\text{Adv}_{\mathcal{B}_2}^{\text{DL}}(\kappa) \geq \text{accept}(\text{Fork}^{\text{SIM}}) - 2B \quad [\text{their (12)}] \quad (9)$$

The three bad events ruled out in GAME 2, and the bounds [14] gives for them, are the following.

bad event	bound	contribution to Equation (9)
ROCollision	$q^2/2p$	$2 \cdot q^2/2p$
$H_{\text{sig}}\text{Guess}$	q/p	$2 \cdot q/p$
FromInvalid _k	q/p	$2 \cdot q/p$
total	$B = \frac{q^2}{2p} + \frac{2q}{p}$	$2B$

The same three quantities occur in Equation (8) and in Equation (9), once in the former and twice in the latter; this is the whole of the arithmetic behind the coefficient $2q$ that appears in Theorem 17. That coefficient is therefore conditional on the correctness of Equation (9), factor 2 included; we do not audit the argument establishing it. Equation (9) is stated in [14], and we reproduce it rather than re-derive it; the factor 2 there reflects that the forking algorithm runs **SIM** twice, so that each bad event is discounted for both runs, and we offer that as an explanation of the shape of their inequality rather than as an independent proof of it.

In Equation (9) we have written B where [14] subtracts twice the three probabilities themselves. This is a weakening, and a legitimate one: the sum of those probabilities is at most B , so that subtracting $2B$ yields a lower bound on $\text{Adv}_{\mathcal{B}_2}^{\text{DL}}$ no larger than theirs, which is the direction we use.

We state Equation (8) as an inequality, where [14] writes a chain of equalities. The difference of the two winning probabilities is bounded by the probability of the union of the three events ruled out in **GAME 2**, hence by their sum, and each of those three probabilities is in turn introduced there as a maximum. Both of the uses we make of Equation (8), in Equation (13) and through Equation (9), require an upper bound.

The proof also uses the identification

$$\Pr[W_2] = \text{accept}(\text{SIM}), \quad (10)$$

which is stated in [14] without argument and which we import as the fifth of the relations we take as given: establishing it would require comparing the two execution distributions in full, which is an audit of the simulator and outside the scope of this paper. What follows is an informal explanation of the intended identification, not a proof of it. If $\mathcal{A}$ wins **GAME 2**, then it has output a valid forgery (m^*, σ^*) with $m^* \notin Q_m$, and **SIM** looks up the entry of its H_{sig} table indexed by (pk, m^*, R^*) and returns its position $I > 0$; if that point had not been queried by $\mathcal{A}$, **SIM** makes the query itself before looking it up, and the summand 1 in $q = q_H + q_S + 1$ is the index reserved for this. Conversely, **SIM** returns $I > 0$ only after checking that $m^* \notin Q_m$ and that $\text{Verify}(m^*, \sigma^*, \text{pk}) = 1$, which is exactly the winning condition of **GAME 2**, so it cannot accept an output that fails verification. The two abort on the same three events, so the abort conditions coincide as well.

Two conventions on resources are in force. The quantities q_H and q_S are fixed a priori upper bounds on the number of queries, not the number actually made: the general forking lemma requires a fixed vector length q , and if $\mathcal{A}$ makes fewer queries the simulator leaves the remaining entries unused. And, as in [14], no

running times are attached to $\mathcal{B}_1$ and $\mathcal{B}_2$; we preserve that omission, and the numerical estimates of Section 7.3 therefore rest on an additional resource model, stated there, in which $\mathcal{B}_2$ performs at most q group operations.

At the end of the proof, the reader is instructed to combine Equation (1) with Equations (6) to (9) in order to obtain Equation (4). We carry this combination out.

5.2 The bound that follows

Theorem 17. *Let $\mathcal{A}, \mathcal{B}_1, \mathcal{B}_2$ and A, B, C be as above. Then Equations (6) to (10) together with Equation (1) imply*

$$\mathbf{Adv}_{\mathcal{A}, \text{Sparkle}+}^{\text{TS-EUF-CMA}}(\kappa) \leq \sqrt{\frac{q^2}{4p^2} + A + 2qB} + \frac{q}{2p} + B + C, \quad (11)$$

and hence also the weaker bound

$$\mathbf{Adv}_{\mathcal{A}, \text{Sparkle}+}^{\text{TS-EUF-CMA}}(\kappa) \leq \sqrt{A + 2qB} + \frac{q}{p} + B + C. \quad (12)$$

Proof. Write $\gamma = \Pr[W_2]$. By Equation (6) and the triangle inequality,

$$\mathbf{Adv}_{\mathcal{A}, \text{Sparkle}+}^{\text{TS-EUF-CMA}}(\kappa) = \Pr[W_0] \leq |\Pr[W_0] - \Pr[W_1]| + |\Pr[W_1] - \Pr[W_2]| + \Pr[W_2],$$

so that, by Equation (7) and Equation (8),

$$\mathbf{Adv}_{\mathcal{A}, \text{Sparkle}+}^{\text{TS-EUF-CMA}}(\kappa) \leq C + B + \gamma. \quad (13)$$

It remains to bound γ . By Equation (10), $\gamma = \text{accept}(\text{SIM})$, and by Equation (9) the quantity $F = \text{accept}(\text{Fork}^{\text{SIM}})$ satisfies $F \leq \mathbf{Adv}_{\mathcal{B}_2}^{\text{DL}}(\kappa) + 2B$. Since the two random oracles of $\text{Sparkle}+$ have codomain $\mathbb{Z}_p$, Lemma 14 applies with $N = p$ and gives

$$\gamma \leq \frac{q}{2p} + \sqrt{\frac{q^2}{4p^2} + qF} \leq \frac{q}{2p} + \sqrt{\frac{q^2}{4p^2} + q\mathbf{Adv}_{\mathcal{B}_2}^{\text{DL}}(\kappa) + 2qB},$$

where the second inequality uses the monotonicity of the square root. Substituting into Equation (13) and recalling $A = q\mathbf{Adv}_{\mathcal{B}_2}^{\text{DL}}(\kappa)$ yields Equation (11). Applying $\sqrt{x+y} \leq \sqrt{x} + \sqrt{y}$ with $x = q^2/4p^2$ and $y = A + 2qB$, and adding the resulting $q/2p$ to the one already present, yields Equation (12). $\square$

Corollary 18. *If q and n are polynomially bounded in κ , if $\mathcal{B}_1$ and $\mathcal{B}_2$ are polynomial-time under those bounds, if $1/p$ is negligible in κ , and if $\mathbf{Adv}_{\mathcal{B}_2}^{\text{DL}}$ and $\mathbf{Adv}_{\mathcal{B}_1, \text{DS}}^{\text{EUF-CMA}}$ are negligible, then the right-hand side of Equation (12) is negligible; assuming the correctness of the imported reductions, $\text{Sparkle}+$ is statically secure.*

Proof. Each of A , C , $2qB$ and q/p is a polynomially bounded multiple of a negligible quantity, and B is a sum of two such multiples of $1/p$; all are therefore negligible, as is any finite sum of them. The square root of a non-negative negligible function ν is negligible as well: given $c > 0$, negligibility of ν with exponent $2c$ gives $\nu(\kappa) < \kappa^{-2c}$ for all large κ , whence $\sqrt{\nu(\kappa)} < \kappa^{-c}$. The sum of the four terms is therefore negligible. $\square$

5.3 Comparison with the stated bound

Equation (12) differs from Equation (4) in three distinct algebraic respects.

- *Placement.* The quantities B and C arise from the two game hops Equation (7) and Equation (8), both of which precede the rewinding. In the direct combination of Equations (6) to (10) they are added after $\Pr[W_2]$ has been bounded, and therefore appear *outside* the square root. In Equation (4) both appear inside it. We do not claim that no valid bound can carry them inside a root, only that this one does not.
- *The coefficient of B inside the root.* The only occurrence of B inside the root comes from Equation (9), and it enters multiplied by $2q$: by the factor 2 of Equation (9) itself, and by the factor q that Lemma 14 introduces. The stated bound carries B with coefficient 1.
- *The forking term.* Inverting Equation (1) contributes $\frac{q}{2p}$ outside the root together with $\frac{q^2}{4p^2}$ inside it; the relaxation Equation (3) combines the two into a single additive $\frac{q}{p}$. Neither occurs in Equation (4).

A mismatch of multiplicities is visible without performing the derivation at all: the game hops contribute B once and C once, whereas Equation (4) carries B once and C twice. This observation is suggestive but not by itself conclusive, since a bound may always be weakened by enlarging one of its terms.

It is also not enough to observe that our bound is, at some parameters, numerically larger than the stated one. From a derivation of $\text{Adv} \leq U$ it does not follow that a smaller V cannot be derived from the same premises by a different route: our U might simply fail to be tight. To show that Equation (4) does not follow from Equations (6) to (10) and Equation (1), one has to exhibit values of the quantities involved that satisfy all of those inequalities and violate Equation (4). Such values exist.

Proposition 19. *There are values of $q, p, n, \text{Adv}_{\mathcal{B}_2}^{\text{DL}}, \text{Adv}_{\mathcal{B}_1, \text{DS}}^{\text{EUF-CMA}}, \Pr[W_0], \Pr[W_1], \Pr[W_2]$ and F , all probabilities lying in $[0, 1]$, which satisfy Equations (6) to (10) and Equation (1) and for which*

$$\text{Adv}_{\mathcal{A}, \text{Sparkle}^+}^{\text{TS-EUF-CMA}}(\kappa) > \sqrt{A + B + 2C}.$$

Consequently Equation (4) is not a consequence of those inequalities.

Proof. Write a_{DL} and a_{DS} for the scalars occupying the formal positions of $\text{Adv}_{\mathcal{B}_2}^{\text{DL}}$ and $\text{Adv}_{\mathcal{B}_1, \text{DS}}^{\text{EUF-CMA}}$. Take for p the order of the `secp256k1` group, a 256-bit prime, and $q = 2^{60}$, $n = 2^{10}$, any t with $1 \leq t < n$, and $a_{\text{DL}} = a_{\text{DS}} = q^2/p$, so that $A = q^3/p$, $B = q^2/2p + 2q/p$ and $C = nq^2/p$. Put

$$F = \text{Adv}_{\mathcal{B}_2}^{\text{DL}} + 2B, \quad \gamma = \frac{q}{2p} + \sqrt{\frac{q^2}{4p^2} + qF},$$

and set $\Pr[W_2] = \gamma$, $\Pr[W_1] = \gamma + B$, $\Pr[W_0] = \gamma + B + C$, with $\text{Adv}_{\mathcal{A}, \text{Sparkle}^+}^{\text{TS-EUF-CMA}}(\kappa) = \Pr[W_0]$. All these quantities lie in $[0, 1]$. Then Equation (6) and Equation (10)

hold by construction, and Equation (7), Equation (8) and Equation (9) hold with equality. Because they hold with equality, the assignment satisfies equally the stronger forms in which [14] displays them, namely Equation (8) as a chain of equalities and Equation (9) with the three bad-event probabilities appearing individually, taken here to be $q^2/2p$, q/p and q/p ; the weakenings adopted in Section 5 have therefore not enlarged the feasible set in any way the countermodel exploits. Finally γ is by construction the positive root of $\gamma^2 - \frac{q}{p}\gamma - qF = 0$, so that $F = \gamma^2/q - \gamma/p$ and Equation (1) holds with equality as well. For the comparison, note that $2qB = 2q(\frac{q^2}{2p} + \frac{2q}{p}) = \frac{q^3}{p} + \frac{4q^2}{p} = A + \frac{4q^2}{p} > A$, so that $\Pr[W_0] > \sqrt{A + 2qB} > \sqrt{2A}$. It therefore suffices that $2A > A + B + 2C$, that is $A > B + 2C$; multiplying by $p/q > 0$ this reads $q^2 > \frac{q}{2} + 2 + 2nq$, which holds at $q = 2^{60}$, $n = 2^{10}$. Hence $\Pr[W_0] > \sqrt{A + B + 2C}$, with no appeal to numerical evaluation. All the assigned values lie in $[0, 1]$, since $A + 2qB < 2^{-74}$ and $B + C < 2^{-125}$ give $\Pr[W_0] < 2^{-37} + 2^{-125} < 1$. $\square$

The construction proves more than non-implication at one parameter point: the same assignment is available throughout the region in which the quantities it produces are probabilities, and it shows that Equation (11) cannot be improved without new information.

Proposition 20. *Regard Equations (6) to (10) and Equation (1) as constraints on real variables w_0, w_1, w_2, f standing for $\Pr[W_0], \Pr[W_1], \Pr[W_2]$ and $\text{accept}(\text{Fork}^{\text{SIM}})$. Let $q, p, \text{Adv}_{\mathcal{B}_2}^{\text{DL}}$ and C be such that $f := \text{Adv}_{\mathcal{B}_2}^{\text{DL}} + 2B$ and the right-hand side U of Equation (11) both lie in $[0, 1]$. Then the constraint system admits an assignment of those variables, all lying in $[0, 1]$, for which $w_0 = U$. Consequently, if V is any expression in $q, p, \text{Adv}_{\mathcal{B}_2}^{\text{DL}}$ and C such that those constraints imply $w_0 \leq V$, then $U \leq V$. Equivalently $U = \sup\{w_0 : \text{the constraints hold}\}$, so that Equation (11) is the pointwise least scalar upper bound implied by the constraint system. The statement is one of algebraic optimality relative to the listed relations, and not of tightness of the reduction: we exhibit no algorithm *SIM* whose acceptance and forking probabilities are the assigned values, and the forking lemma gives an inequality which an actual simulator need not saturate.*

Proof. Take the assignment of the proof of Proposition 19, with the parameters left free: put $f = \text{Adv}_{\mathcal{B}_2}^{\text{DL}} + 2B$, let γ be the larger root of $\gamma^2 - \frac{q}{p}\gamma - qf = 0$, and set $w_2 = \gamma$, $w_1 = \gamma + B$, $w_0 = \gamma + B + C$. As there, Equation (7), Equation (8), Equation (9) and Equation (1) hold with equality and Equation (6), Equation (10) by construction, while $\gamma = \frac{q}{2p} + \sqrt{\frac{q^2}{4p^2} + qf}$ gives $w_0 = U$ exactly. The hypothesis places all the quantities in $[0, 1]$. For the consequence: if the constraints imply $w_0 \leq V$, they imply it in particular at this assignment, where $w_0 = U$; hence $U \leq V$. $\square$

Proposition 19 is the special case obtained by evaluating U and $\sqrt{A + B + 2C}$ at the parameters given there. Improving Equation (11) therefore requires more than better algebra: it requires a tighter game transition, a sharper forking argument, or cryptographic information that Equations (6) to (10) do not carry.

What Proposition 19 establishes is non-implication at the level of those scalar inequalities; it exhibits no adversary against Sparkle+ realising the values constructed, and we make no such claim. The parameters are chosen only to make the inequality hold; the proposition concerns the scalar system and not any instantiation of the scheme.

We emphasise that neither expression is uniformly smaller than the other: whether Equation (12) is tighter or looser than Equation (4) depends on the parameters, and we determine the region in which each holds in Section 7.

6 The Adaptive Bound

The second result of [14] asserts the adaptive security of Sparkle+ against up to $t/2$ corruptions, under the AOMDL assumption and the EUF-CMA security of DS, in the random oracle model. Both the quoted statement and the bound derived below use the fixed- t reading of Section 3, in which n and $t + 1$ are inputs to the game. The corruption threshold is $\lfloor t/2 \rfloor$ rather than t because the reduction rewinds the adversary once, and the corruptions performed in the two runs must together not exceed the $t + 1$ -AOMDL oracle allows.

Theorem 21 ([14, Theorem 2]). *Let GrGen be a group generator and DS be a digital signature scheme. For any PPT adversary $\mathcal{A}$ for the adaptive unforgeability game $\text{Game}_{\mathcal{A}, \text{Sparkle}+}^{\text{adp-TS-EUF-CMA}}(\kappa, \frac{1}{2})$, there exists a PPT adversary $\mathcal{B}'_1$ for the game $\text{Game}_{\mathcal{B}'_1, \text{DS}}^{\text{EUF-CMA}}(\kappa)$ making at most q_S queries to its signing oracle, and a PPT adversary $\mathcal{B}'_2$ for the game $\text{Game}_{\mathcal{B}'_2}^{t+1\text{-AOMDL}}(\kappa)$ such that*

$$\text{Adv}_{\mathcal{A}, \text{Sparkle}+}^{\text{adp-TS-EUF-CMA}}(\kappa, \frac{1}{2}) \leq \sqrt{q \text{Adv}_{\mathcal{B}'_2}^{t+1\text{-AOMDL}}(\kappa) + \frac{q^2}{2p} + \frac{5q}{p} + \frac{n^2}{(n - \frac{t}{2})} \text{Adv}_{\mathcal{B}'_1, \text{DS}}^{\text{EUF-CMA}}(\kappa)} \quad (14)$$

where n is the number of signers, the game permits $\lfloor t/2 \rfloor$ corruptions, p is the group size, and $q = q_H + q_S + 1$.

In analogy with Equation (5) we abbreviate

$$A' = q \cdot \text{Adv}_{\mathcal{B}'_2}^{t+1\text{-AOMDL}}(\kappa), \quad B' = \frac{q^2}{2p} + \frac{5q}{p}, \quad C' = \frac{n^2}{n - t/2} \cdot \text{Adv}_{\mathcal{B}'_1, \text{DS}}^{\text{EUF-CMA}}(\kappa), \quad (15)$$

so that the right-hand side of Equation (14) is $\sqrt{A' + B' + C'}$. Note that, unlike in the static case, the DS term occurs inside the square root with coefficient 1 and not 2.

6.1 The relations supplied by the source proof

Throughout this section we write $c := \lfloor t/2 \rfloor$ for the number of corruptions that the game of Table 4 permits when $\text{frac} = 1/2$; [14] writes $t/2$, the two agreeing

whenever t is even, and we keep their notation inside quoted statements and use c in our own.

The proof of Theorem 21 follows the same three-game pattern as that of Theorem 16, with the simulator now denoted SIM' and equipped with the discrete logarithm oracle of the $t+1$ -AOMDL game, which answers at most t valid queries. Two differences are relevant here. First, the reduction $\mathcal{B}'_1$ carries the coefficient $\frac{n^2}{n-t/2}$, while the static one carries n . As there, it embeds its EUF-CMA challenge key at a party τ drawn uniformly among the parties honest at key generation, the embedded key being distributed exactly as an honestly generated auxiliary public key, so that τ is hidden from $\mathcal{A}$; and it succeeds only if the forgery is produced for τ . In the adaptive experiment τ may afterwards be corrupted, since the adversary is not confined to the set it declared at the outset, and $\mathcal{B}'_1$ aborts if that happens. The coefficient established in [14] from these two sources of loss is the one appearing in Equation (14); we import Equation (17) as it stands, with that coefficient, and do not re-derive it. Since $c = \lfloor t/2 \rfloor \leq t/2$ we have $\frac{n^2}{n-c} \leq \frac{n^2}{n-t/2}$, so keeping the source's expression is the conservative choice when t is odd. Second, GAME 2 rules out six bad events instead of three. Of the three additional ones, $\text{H}_{\text{cm}}\text{Guess}$ arises from the corruption oracle, which must hand over a nonce for a commitment whose preimage the adversary may already have queried, while $\text{H}_{\text{sig}}\text{GuessOneHon}$ and $\text{H}_{\text{sig}}\text{GuessLastHonCor}$ are consequences of the delayed programming of H_{sig} ; see Remark 25 below.

Writing again W_i for the event that $\mathcal{A}$ wins GAME i , the proof establishes the following relations, whose numbering in the June 2025 revision of [14], that is v4 of Table 8, we recall in brackets.

$$\text{Adv}_{\mathcal{A}, \text{Sparkle}^+}^{\text{adv-TS-EUF-CMA}}(\kappa, \text{frac} = \tfrac{1}{2}) = \Pr[W_0] \quad [\text{their (13)}] \quad (16)$$

$$|\Pr[W_0] - \Pr[W_1]| \leq C' \quad [\text{their (14)}] \quad (17)$$

$$|\Pr[W_1] - \Pr[W_2]| \leq B' \quad [\text{their (15)}] \quad (18)$$

$$\text{Adv}_{\mathcal{B}'_2}^{t+1\text{-AOMDL}}(\kappa) \geq \text{accept}(\text{Fork}^{\text{SIM}'}) - 2B' \quad [\text{their (16)}] \quad (19)$$

together with the identification

$$\Pr[W_2] = \text{accept}(\text{SIM}'), \quad (20)$$

which we import on the same footing as Equation (10), and for which [14] gives the same reason, with three adaptive-specific additions. The corruption oracle is answered by SIM' exactly as GAME 2 answers it, so it contributes no discrepancy between the two; the two abort on the same six events rather than the same three; and the winning condition of the adaptive game carries the additional requirement that the number of corruptions not exceed c , which SIM' enforces through the same oracle. In the forward direction, a win in GAME 2 yields a valid forgery whose H_{sig} point SIM' locates, or inserts and then locates, returning its index $I > 0$; in the converse direction SIM' returns $I > 0$ only after the same checks that the game makes.

The factor 2 in Equation (19) accounts for the two runs of SIM' performed by the forking algorithm: each of the six bad events whose probabilities are bounded by B' in Equation (18) is subtracted twice. As in the static case, Equation (19) is established in [14] and taken here as an input; the two runs explain the shape of the factor and are not offered as an independent derivation of it, the second run sharing a prefix with the first and being launched at a data-dependent index. As in the static case, Equation (18) is stated here as an inequality where [14] writes an equality, and Equation (19) is stated with B' in place of the six probabilities that [14] subtracts individually; both are weakenings in the direction we use, for the reasons given after Equation (9), and neither has any consequence for the derivation. The six events, their bounds and their two occurrences are collected in Table 6. The reader is then instructed to combine Equation (1) with Equations (16) to (19).

bad event ruled out in GAME 2	bound	contribution to Equation (19)
R0Collision	$q^2/2p$	twice
$\text{H}_{\text{sig}}\text{Guess}$	q/p	twice
FromInvalid_k	q/p	twice
$\text{H}_{\text{cm}}\text{Guess}$	q/p	twice
$\text{H}_{\text{sig}}\text{GuessOneHon}$	q/p	twice
$\text{H}_{\text{sig}}\text{GuessLastHonCor}$	q/p	twice
total	$B' = \frac{q^2}{2p} + \frac{5q}{p}$	$2B'$

Table 6. The six events ruled out in GAME 2 of [14, Appendix B]. The first three are those listed in Section 5; of the other three, $\text{H}_{\text{cm}}\text{Guess}$ was already present in the 2024 version, which ruled out four events in all, while the last two were introduced by the repair of the simulation (Remark 25).

6.2 The bound that follows

Theorem 22. *Let $\mathcal{A}, \mathcal{B}'_1, \mathcal{B}'_2$ and $\mathcal{A}', \mathcal{B}', \mathcal{C}'$ be as above. Then Equations (16) to (20) together with Equation (1) imply*

$$\text{Adv}_{\mathcal{A}, \text{Sparkle}^+}^{\text{adp-TS-EUF-CMA}}(\kappa, \frac{1}{2}) \leq \sqrt{\frac{q^2}{4p^2} + A' + 2qB'} + \frac{q}{2p} + B' + C', \quad (21)$$

and hence also

$$\text{Adv}_{\mathcal{A}, \text{Sparkle}^+}^{\text{adp-TS-EUF-CMA}}(\kappa, \frac{1}{2}) \leq \sqrt{A' + 2qB'} + \frac{q}{p} + B' + C'. \quad (22)$$

Proof. Write $\gamma' = \Pr[W_2]$. By Equation (16) and the triangle inequality,

$$\text{Adv}_{\mathcal{A}, \text{Sparkle}^+}^{\text{adp-TS-EUF-CMA}}(\kappa, \frac{1}{2}) = \Pr[W_0] \leq |\Pr[W_0] - \Pr[W_1]| + |\Pr[W_1] - \Pr[W_2]| + \Pr[W_2],$$

so that, by Equation (17) and Equation (18),

$$\text{Adv}_{\mathcal{A}, \text{Sparkle+}}^{\text{adp-TS-EUF-CMA}}(\kappa, \frac{1}{2}) \leq C' + B' + \gamma'. \quad (23)$$

Since $\gamma' = \text{accept}(\text{SIM}')$ by Equation (20) and, by Equation (19), the quantity $F' = \text{accept}(\text{Fork}^{\text{SIM}'})$ satisfies $F' \leq \text{Adv}_{\mathcal{B}'_2}^{t+1\text{-AOMDL}}(\kappa) + 2B'$, Lemma 14 applied with $N = p$ gives

$$\gamma' \leq \frac{q}{2p} + \sqrt{\frac{q^2}{4p^2} + qF'} \leq \frac{q}{2p} + \sqrt{\frac{q^2}{4p^2} + A' + 2qB'}.$$

Substituting into Equation (23) yields Equation (21); applying $\sqrt{x+y} \leq \sqrt{x} + \sqrt{y}$ with $x = q^2/4p^2$ and $y = A' + 2qB'$ yields Equation (22). $\square$

Corollary 23. *If q, n and t are polynomially bounded in κ , if $1/p$ is negligible in κ , and if $\text{Adv}_{\mathcal{B}'_2}^{t+1\text{-AOMDL}}$ and $\text{Adv}_{\mathcal{B}'_1, \text{DS}}^{\text{EUF-CMA}}$ are negligible, then the right-hand side of Equation (22) is negligible; assuming the correctness of the imported reductions, *Sparkle+* is adaptively secure against up to $c = \lfloor t/2 \rfloor$ corruptions.*

Proof. As for Corollary 18. The only point requiring comment is the multiplier of the DS advantage: since $t < n$ we have $n - t/2 > n/2$ and therefore $\frac{n^2}{n-t/2} < 2n$, which is polynomially bounded; so C' is a polynomially bounded multiple of a negligible quantity, hence negligible. Each of $A', B', 2qB'$ and q/p is negligible for the same reasons as in the static case, and the argument bounding $\sqrt{A' + 2qB'}$ is unchanged. $\square$

As in the static case, and as in [14], no running times are attached to $\mathcal{B}'_1$ and $\mathcal{B}'_2$; we preserve that omission, so that the numerical estimates of Section 7.3 rest on the additional resource model declared there.

6.3 Comparison with the stated bound

The same three algebraic effects recur, with A', B', C' in place of A, B, C ; they are not literally the same, since the assumption changes from DL to AOMDL, $B' \neq B$, C' carries the additional adaptive-abort loss, and the coefficient of C' in the stated bound is 1 rather than 2. In the direct combination of Equations (16) to (20) the terms B' and C' are added after $\Pr[W_2]$ has been bounded and therefore appear outside the square root, without our claiming that no other valid relaxation could place them inside one; the only occurrence of B' inside the root carries the coefficient $2q$, from the factor 2 of Equation (19) and the factor q of Lemma 14; and inverting Equation (1) contributes $\frac{q}{2p}$ outside the root together with $\frac{q^2}{4p^2}$ inside it, the relaxation Equation (3) combining the two into a single additive $\frac{q}{p}$. None of these occurs in Equation (14).

One observation of Section 5 does not carry over. There the mismatch of multiplicities was visible without performing any derivation, the DS term appearing with multiplicity 2 where the game hops supply it once. Here Equation (14)

carries C' with multiplicity 1, in agreement with Equation (17): for the DS term only the placement is at issue. The aggregate B' still differs in both respects, appearing once inside the root in Equation (14) and, in the combination, once outside it together with $2qB'$ inside.

That Equation (14) does not follow from Equations (16) to (20) is established as in the static case, and we make the construction explicit.

Proposition 24. *Regard Equations (16) to (20) and Equation (1) as constraints on real variables w'_0, w'_1, w'_2, f' standing for $\Pr[W_0]$, $\Pr[W_1]$, $\Pr[W_2]$ and $\text{accept}(\text{Fork}^{\text{SI}'}))$. The constraint system admits an assignment, all of whose values lie in $[0, 1]$, with $w'_0 > \sqrt{A' + B' + C'}$; hence Equation (14) is not a consequence of those relations.*

Proof. Write a_{AOMDL} and a_{DS} for the scalars occupying the formal positions of the two advantages. Take for p the order of the `secp256k1` group, $q = 2^{60}$, $n = 2^{10}$, $t = 2$, so that $c = 1$, and set $a_{\text{AOMDL}} = a_{\text{DS}} = q^2/p$. Then $A' = q^3/p$, $B' = \frac{q^2}{2p} + \frac{5q}{p}$ and $C' = \frac{n^2}{n-1} \cdot \frac{q^2}{p}$. Put

$$f' = \text{Adv}_{B'_2}^{t+1-\text{AOMDL}} + 2B', \quad \gamma' = \frac{q}{2p} + \sqrt{\frac{q^2}{4p^2} + qf'},$$

and set $w'_2 = \gamma'$, $w'_1 = \gamma' + B'$ and $w'_0 = \gamma' + B' + C'$, all of which lie in $[0, 1]$. Then Equation (16) and Equation (20) hold by definition, while Equation (17), Equation (18) and Equation (19) hold with equality; γ' is the positive root of $\gamma'^2 - \frac{q}{p}\gamma' - qf' = 0$, so Equation (1) holds with equality as well. As in the static case $2qB' = \frac{q^3}{p} + \frac{10q^2}{p} > A'$, so $w'_0 > \sqrt{2A'}$, and it suffices that $A' > B' + C'$; multiplying by p/q this reads $q^2 > \frac{q}{2} + 5 + \frac{n^2}{n-1}q$, which holds at $q = 2^{60}$, $n = 2^{10}$. The assigned values lie in $[0, 1]$ by the same coarse bounds. $\square$

As in the static case this concerns the scalar inequalities only, and the construction is available throughout the region delimited by the probability-range conditions of Proposition 20, not only at the point displayed: the assignment above attains the right-hand side of Equation (21) exactly, so the argument of Proposition 20 carries over, with A', B', C' and the AOMDL advantage in place of their static counterparts: whenever $f' = a_{\text{AOMDL}} + 2B'$ and the right-hand side U' of Equation (21) both lie in $[0, 1]$, the constraint system admits an assignment with $w'_0 = U'$, and Equation (21) is therefore the least bound derivable from Equations (16) to (20) and Equation (1). As there, this is optimality relative to the listed scalar relations and not tightness of the reduction.

Remark 25. The value $B' = q^2/2p + 5q/p$ is that of the current version of [14]. It exceeds by $2q/p$ the value that the same hop carried in the 2024 version, in which GAME 2 ruled out four bad events rather than six; in the two 2023 versions the bounds are asymptotic and no such quantity appears at all (Section 8). The two additional events are $\text{H}_{\text{sig}}\text{GuessOneHon}$ and $\text{H}_{\text{sig}}\text{GuessLastHonCor}$, and they are

the price of the repair described in the introduction: since H_{cm} is programmed only gradually, the programming of H_{sig} must be deferred until the signing session is fully determined, which may happen either in a later second-round query or, in the adaptive case, at a corruption query. Both events are bounded by q/p . We record this to make clear that the increase of B' between versions is accounted for in [14], and is unrelated to the discrepancies discussed above.

7 Comparison of the Two Bounds

7.1 Full Static Case

Throughout this subsection A , B and C are as in Equation (5), and we assume

$$A + B + 2C \leq \frac{1}{4}, \quad (24)$$

a hypothesis satisfied by a wide margin at the parameter sets of Section 7.3, and which we use only to bound the quantity S introduced below.

We want to understand when, in the full static secure case, our derived expression is smaller than the published one, that is to understand when

$$\eta_{\text{orig}} = \sqrt{A + B + 2C} \geq \sqrt{A + 2qB} + B + C + \frac{q}{p} = \eta_{\text{new}}$$

holds, where η_{new} is the right-hand side of Equation (12). We compare first against the relaxed form; the criterion for the exact form itself is recorded in Lemma 28. Writing $a = q/2p$ and $D = A + 2qB$, the two differ by $a + \sqrt{D} - \sqrt{D + a^2}$, a quantity lying in $[0, a]$, so that

$$0 \leq \eta_{\text{new}} - \left(\frac{q}{2p} + \sqrt{\frac{q^2}{4p^2} + A + 2qB} \right) \leq \frac{q}{2p}.$$

The comparison below therefore understates the tightness of our exact bound, and the region in which it is the tighter is if anything slightly larger than the one determined here.

Lemma 26 (Exact condition for the relaxed bound). *Let $S := \sqrt{A + B + 2C} + \sqrt{A + 2qB}$, a quantity that is strictly positive since $B > 0$. Then*

$$\eta_{\text{new}} \leq \eta_{\text{orig}} \iff \left(B + C + \frac{q}{p} \right) \cdot S \leq 2C - (2q - 1)B. \quad (25)$$

Proof. The claim $\eta_{\text{new}} \leq \eta_{\text{orig}}$ is equivalent to $B + C + \frac{q}{p} \leq \sqrt{A + B + 2C} - \sqrt{A + 2qB}$. Multiplying and dividing the right-hand side by the conjugate $S > 0$ gives

$$\sqrt{A + B + 2C} - \sqrt{A + 2qB} = \frac{(A + B + 2C) - (A + 2qB)}{S} = \frac{2C - (2q - 1)B}{S},$$

and multiplying through by S yields (25). Since the left-hand side of (25) is strictly positive, a necessary condition for $\eta_{\text{new}} \leq \eta_{\text{orig}}$ is $2C > (2q - 1)B$: the DS term must dominate the bad-event aggregate by a factor of order q . $\square$

We now show that a sufficient condition for (25) to be valid is to assume that $4qB \leq C$.

Proposition 27. *Assume Equation (24). If $4qB \leq C$, then $\eta_{\text{new}} \leq \eta_{\text{orig}}$.*

Proof. Suppose $4qB \leq C$. Then $2qB \leq C/2 \leq B + 2C$, hence $A + 2qB \leq A + B + 2C$ and therefore

$$S \leq 2\sqrt{A + B + 2C} \leq 2 \cdot \frac{1}{2} = 1$$

by Equation (24). Thus $(B + C + \frac{q}{p})S \leq B + C + \frac{q}{p}$, and it suffices to check

$$B + C + \frac{q}{p} \leq 2C - (2q - 1)B = 2C - 2qB + B,$$

that is, $2qB + \frac{q}{p} \leq C$. Now $\frac{q}{p} \leq B$, since $B = \frac{q^2}{2p} + \frac{2q}{p}$, and $B \leq 2qB$ because $q \geq 1$; hence $\frac{q}{p} \leq 2qB \leq C/2$. This gives $2qB + \frac{q}{p} \leq C$, so that (25) holds and the conclusion follows from Lemma 26. The hypothesis $4qB \leq C$ is sufficient and not necessary: the exact region is the one delimited by (25), of which it is a convenient sub-region. $\square$

The same conjugate manipulation gives the criterion for the exact bounds Equation (11) and Equation (21), which we record so that the comparison is not confined to the relaxed forms.

Lemma 28 (Exact condition for the exact bound). *Write $a = \frac{q}{2p}$, $X = A + B + 2C$ and $Y = A + 2qB + a^2$, and let $S_{\text{ex}} = \sqrt{X} + \sqrt{Y}$. Then the right-hand side of Equation (11) is at most η_{orig} if and only if*

$$(a + B + C) S_{\text{ex}} \leq 2C - (2q - 1)B - a^2. \quad (26)$$

In the adaptive case, with $X' = A' + B' + C'$ and $Y' = A' + 2qB' + a^2$, the corresponding criterion is $(a + B' + C')(\sqrt{X'} + \sqrt{Y'}) \leq C' - (2q - 1)B' - a^2$.

Proof. The claim is $a + B + C + \sqrt{Y} \leq \sqrt{X}$, that is $a + B + C \leq \sqrt{X} - \sqrt{Y} = (X - Y)/S_{\text{ex}}$, and $X - Y = (A + B + 2C) - (A + 2qB + a^2) = 2C - (2q - 1)B - a^2$. Multiplying by $S_{\text{ex}} > 0$ gives (26); the adaptive computation is identical, the coefficient of C' in X' being 1 rather than 2. $\square$

Written out, the sufficient condition of Proposition 27 reads

$$4q \left(\frac{q^2}{2p} + \frac{2q}{p} \right) \leq n \cdot \text{Adv}_{B_1}^{\text{EUF-CMA}},$$

which we instantiate in Section 7.3.

7.2 Adaptive Case with $\lfloor t/2 \rfloor$ Corrupted Parties

The same analysis can be carried out for Theorem 21, with two differences: the DS term occurs in Equation (14) with coefficient 1 rather than 2, and the smallness hypothesis must be slightly strengthened. Let A', B', C' be as in Equation (15), assume

$$A' + B' + C' \leq \frac{1}{16}, \quad (27)$$

and set

$$\eta'_{\text{orig}} = \sqrt{A' + B' + C'}, \quad \eta'_{\text{new}} = \sqrt{A' + 2qB'} + B' + C' + \frac{q}{p},$$

the latter being the right-hand side of Equation (22).

Lemma 29 (Exact condition for the relaxed bound). *Let $S' := \sqrt{A' + B' + C'} + \sqrt{A' + 2qB'}$, again strictly positive. Then*

$$\eta'_{\text{new}} \leq \eta'_{\text{orig}} \iff \left(B' + C' + \frac{q}{p}\right) \cdot S' \leq C' - (2q - 1)B'. \quad (28)$$

Proof. Identical to that of Lemma 26: the claim is equivalent to $B' + C' + \frac{q}{p} \leq \sqrt{A' + B' + C'} - \sqrt{A' + 2qB'}$, and multiplying and dividing the right-hand side by the conjugate $S' > 0$ turns its numerator into $(A' + B' + C') - (A' + 2qB') = C' - (2q - 1)B'$. As in the static case, $C' > (2q - 1)B'$ is necessary. $\square$

Proposition 30. *Assume Equation (27). If $4qB' \leq C'$, then $\eta'_{\text{new}} \leq \eta'_{\text{orig}}$.*

Proof. From $4qB' \leq C'$ we get $2qB' \leq C'/2 \leq B' + C'$, hence $A' + 2qB' \leq A' + B' + C'$ and therefore $S' \leq 2\sqrt{A' + B' + C'} \leq 2 \cdot \frac{1}{4} = \frac{1}{2}$ by Equation (27). Since $\frac{q}{p} \leq B'$, the left-hand side of Equation (28) is at most $\frac{1}{2}(2B' + C') = B' + \frac{C'}{2}$, while its right-hand side is

$$C' - (2q - 1)B' = C' - 2qB' + B' \geq C' - \frac{C'}{2} + B' = \frac{C'}{2} + B'.$$

Hence Equation (28) holds, and the conclusion follows from Lemma 29. $\square$

Written out, the sufficient condition reads

$$4q \left(\frac{q^2}{2p} + \frac{5q}{p} \right) \leq \frac{n^2}{n - t/2} \cdot \text{Adv}_{B'_1, \text{DS}}^{\text{EUF-CMA}},$$

and differs from its static counterpart in the value of the collision term, in the coefficient of the DS advantage, which is here larger than n by a factor $n/(n - t/2) \in [1, 2)$, the value 2 not being attained since $t < n$, and in the smallness hypothesis, which has to be Equation (27) rather than Equation (24).

7.3 Concrete Parameters

The criterion $4qB \leq C$ of Proposition 27 is a condition on the relative sizes of the bad-event aggregate and of the DS advantage, and the choice of the auxiliary signature scheme is the principal factor separating the two regimes below, though the condition also depends on q , p , n and on the resource bounds of $\mathcal{B}_1$. We instantiate it.

The computation that follows is an illustrative order-of-magnitude comparison, not a complete concrete-security estimate. We identify all the relevant resource budgets, the oracle-query counts of $\mathcal{A}$, the number q_{DS} of queries made to the random oracle of DS when that scheme is instantiated in the random oracle model, and the group-operation counts of $\mathcal{B}_1$ and $\mathcal{B}_2$ and of the reductions internal to DS, with the single parameter q , so that in particular $q_{\text{DS}} \leq q$; we suppress constants and lower-order terms; and we substitute for each advantage its generic upper-bound proxy. This last convention is the one to bear in mind when reading the table: the criterion $4qB \leq C$ requires a lower bound on C , whereas a security reduction supplies an upper bound, and the adversary $\mathcal{B}_1$ that the threshold reduction actually produces may have advantage far below its proxy. The table therefore shows how the two expressions compare when the advantages attain their proxies, not which is smaller for the adversaries the reductions actually produce. None is implied by the theorems, which attach no running times to $\mathcal{B}_1$ and $\mathcal{B}_2$; they are conventions adopted to isolate the different algebraic placement of the terms.

Recall that

$$A = q \cdot \text{Adv}_{\mathcal{B}_2}^{\text{DL}}(\kappa), \quad B = \frac{q^2}{2p} + \frac{2q}{p}, \quad C = n \cdot \text{Adv}_{\mathcal{B}_1, \text{DS}}^{\text{EUFCMA}}(\kappa).$$

Fix accordingly a group of prime order $p \approx 2^{256}$, for instance `secp256k1`, and set $\text{Adv}_{\mathcal{B}_2}^{\text{DL}} \approx q^2/p$, hence $A \approx q^3/p$. Two consequences are worth noting before the computation. First, $2qB \approx q^3/p \approx A$, so that the term $2qB$ inside our square root is of the same order as A and contributes a factor $\sqrt{2}$ rather than a change of magnitude. Second, the exact and relaxed forms Equation (11) and Equation (12) agree to within a fraction of a bit, since they differ only by $q/2p$ and by the term $q^2/4p^2$ inside the root, both negligible here. Finally, the hypotheses Equation (24) and Equation (27) hold with an enormous margin at every parameter set below: the largest value of $A + B + 2C$ occurring in the table is of the order of 2^{-23} , and since $C' = \alpha C < 2C$ with α as in the adaptive comparison below, and $B' < 3B$, we get $A' + B' + C' < A' + 3B + 2C$, which under the identification $A' = A$ is again of that order. For DS we consider the regime proposed in the instantiation [14]: its EUF-CMA security is itself proven by rewinding, so that $\text{Adv}^{\text{EUF-CMA}} \approx \sqrt{q} \cdot \text{Adv}^{\text{DL}} \approx \sqrt{q^3/p}$.

Since DS is a Schnorr signature, C dominates both A and B , so that $\eta_{\text{orig}} \approx \sqrt{2C}$ while $\eta_{\text{new}} \approx C$: placing the DS term outside the square root removes a square root from the dominant term, and the gain, by which we mean $\log_2(\eta_{\text{orig}}/\eta_{\text{new}})$, is of the order of half the bit size of C , between 12 and 39 bits over the range considered.

DS	q	n	published η_{orig}	derived η_{new}
Schnorr	2^{30}	2^7	$2^{-37.5}$	$2^{-76.0}$
Schnorr	2^{45}	2^7	$2^{-26.2}$	$2^{-53.5}$
Schnorr	2^{45}	2^{10}	$2^{-24.8}$	$2^{-50.5}$
Schnorr	2^{60}	2^{10}	$2^{-13.5}$	$2^{-28.0}$
Schnorr	2^{60}	2^{14}	$2^{-11.5}$	$2^{-24.0}$

Table 7. The published expression and the derived one, for $p \approx 2^{256}$, the entries being computed with $p = 2^{256}$, from which the order of `secp256k1` differs by less than 2^{-127} , $\text{Adv}^{\text{DL}} \approx q^2/p$. At the proxy values, the condition $4qB \leq C$ of Proposition 27 holds in every row.

One consequence is worth recording. Under the conventions declared above, and in particular when the advantages attain their proxies, the derived expression is substantially the smaller at every Schnorr parameter set listed in Table 7; for a $\mathcal{B}_1$ of smaller advantage the comparison can reverse, in the manner characterised exactly by Lemma 26.

The same computation for Theorem 21 differs in three respects. It rests on two further proxies: that $\mathcal{B}_1$ and $\mathcal{B}'_1$ are modelled by the same EUF-CMA advantage, so that $C' = \alpha C$, and that $A' = A$, although no reduction relates the AOMDL and DL advantages. The replacement of B by $B' = q^2/2p + 5q/p$ changes no entry of Table 7 by more than a fraction of a bit. Write $\alpha = n/(n - t/2) \in [1, 2)$, so that $C' = \alpha C$ under the model above. In Table 7 the static gain is $-\frac{1}{2} \log_2 C + \frac{1}{2}$, coming from $\eta_{\text{orig}} \approx \sqrt{2C}$ against $\eta_{\text{new}} \approx C$, while the adaptive gain is $-\frac{1}{2} \log_2(\alpha C)$, since the DS term enters Equation (14) with coefficient 1 rather than 2. The difference is $\frac{1}{2} + \frac{1}{2} \log_2 \alpha$, so the adaptive gains are smaller than the static ones by between half a bit and one bit, and the qualitative picture is unchanged. We also set $A' = A$ in this comparison, which is a further modelling assumption: the two are advantages against different problems.

8 The Stated Bounds Across Versions

The ePrint record of [14] consists of the original 2023 submission and three subsequent revisions, giving four archived snapshots. They record a substantial amount of work: the scheme itself changed, from `Sparkle` to `Sparkle+`, in response to [4]; the bounds passed from asymptotic to concrete; and the static simulation was repaired, the inconsistency it addresses having been identified in [11]. We record the history because it bears on the discrepancies of Section 5 and Section 6, and in particular because it locates the first appearance of the additional forking term of Lemma 14 in the direct inversion of the forking inequality that the proofs use. Each snapshot is cited individually below, so that every formula quoted can be checked against the archived document rather than against the current one.

The two 2023 versions. In v1 and v2 the scheme is `Sparkle`, there is no auxiliary signature scheme, and consequently no EUF-CMA term. The bounds are asymp-

version	date	forking lemma	stated bound of Theorem 1
v1 [15]	2023-03-27	local [6]	$\sqrt{q\text{Adv}^{\text{DL}} + \text{negl}(\lambda)}$
v2 [16]	2023-06-23	local [6]	$\sqrt{q\text{Adv}^{\text{DL}} + \text{negl}(\kappa)}$
v3 [17]	2024-05-27	general [7]	$\sqrt{q\text{Adv}^{\text{DL}} + \frac{q^2+4q}{p} + 2n\text{Adv}^{\text{EUF-CMA}}}$
v4 [14]	2025-06-03	general [7]	$\sqrt{q\text{Adv}^{\text{DL}} + \frac{q^2}{2p} + \frac{2q}{p} + 2n\text{Adv}^{\text{EUF-CMA}}}$

Table 8. The statement of the full static security theorem across the four archived versions of [14]. All four set $q = q_H + q_S + 1$, so that the coefficients compared here are coefficients of the same quantity. The scheme is **Sparkle** in v1 and v2 and **Sparkle+** from v3 on. Adversary subscripts and security-parameter arguments of the advantages are suppressed, and each occurrence of negl may denote a different negligible function; the security parameter is written λ in v1 and κ thereafter, and neither is to be confused with the Lagrange coefficients λ_i of Definition 3.

otic: all concrete quantities are absorbed into negl , and the step displayed in the source is $\frac{1}{q}(\text{Adv})^2 \leq \text{Adv}^{\text{DL}} + \text{negl}$. Solving this gives $\text{Adv} \leq \sqrt{q\text{Adv}^{\text{DL}} + q\text{negl}}$, and since q is polynomial the factor q on the second summand is absorbed into another negligible function; the entry in Table 8 is normalised in this way rather than quoted verbatim. The reduction reprograms H_{sig} at a single point and invokes the *local* forking lemma of Bellare, Dai and Li [6] and displays the instantiated inequality $\text{accept}(\text{Fork}) \geq \text{accept}(\text{SIM})^2/q$, without the subtractive correction that Equation (1) carries. We report that displayed step as it is written; a full comparison of the two forking frameworks would additionally require matching their fork-success events and their challenge-resampling conventions, which we do not undertake. At the asymptotic level of these two versions the concrete quantities studied in this note, any polynomially scaled $1/p$ contribution and the game-hop terms themselves, are absorbed into negl .

The 2024 version. In v3 **Sparkle+** and the term $2n\text{Adv}^{\text{EUF-CMA}}$ appear, the bounds become concrete, and the reduction is recast in terms of the *general* forking lemma of Bellare and Neven [7]. In the recast simulation the random-oracle answers are represented by a vector $(h_1, \dots, h_q)$ supplied to the simulator, and the proof introduces a random-oracle collision event bounded by $q^2/2p$, which is the first appearance of that term. The inequality invoked in v3 contains the correction $-\text{accept}(\text{SIM})/|H|$, whereas the one displayed in v1 and v2 carries no analogous term. Inverting the former exactly contributes $\frac{q}{2|H|}$ outside the square root together with $\frac{q^2}{4|H|^2}$ inside it (Lemma 14), and its standard relaxation yields the simpler additive $\frac{q}{|H|}$ (Corollary 15). Neither appears in the statement of v3, nor in that of v4. This is the third of the algebraic differences of Section 5: v3 is the first version in which the term enters the direct inversion, and it is absent from that statement and from every later one. We do not claim that no valid bound could avoid it by some other relaxation, nor do the countermodels of Section 5 and Section 6 isolate it as an independent cause of the failure they

exhibit. We record this as a chronological observation and draw no inference about how the omission arose.

The 2025 version. In v4 the static simulation is repaired as described in the introduction, and the repair is propagated to the $t/2$ adaptive proof. The static simulation is therefore modified; what is unchanged is the numerical form of the static intermediate inequalities: the hop we import as Equation (8) reads $q^2/2p + 2q/p$ in v4, the corresponding inequality of v3 reads the same, and in both versions the extraction step subtracts twice that quantity. Among the displayed aggregates compared here, only the adaptive one changes, growing by $2q/p$ on account of the two events $H_{\text{sig}}\text{GuessOneHon}$ and $H_{\text{sig}}\text{GuessLastHonCor}$ introduced by the repair. The coefficient with which the static hop quantity appears in the *statement*, however, changed: v3 carries $(q^2 + 4q)/p$, which is exactly $2B$, and v4 carries B .

quantity	v3 (2024)	v4 (2025)
static game hop, Equation (8)	B	B
static extraction correction, Equation (9)	$-2B$	$-2B$
B -contribution in static Theorem 1	$2B$	B
adaptive game hop, Equation (18)	$\frac{q^2}{2p} + \frac{3q}{p}$	$\frac{q^2}{2p} + \frac{5q}{p}$
bad-event contribution in adaptive Theorem 2	$\frac{q^2 + 6q}{p}$	$\frac{q^2}{2p} + \frac{5q}{p}$

Table 9. The static and adaptive quantities in v3 [17] and v4 [14]. In the static rows the two displayed inequalities are identical while the statement halves. In the adaptive rows the hop grows by $2q/p$, for the documented reason recalled in Remark 25, and the statement passes from twice the v3 hop to once the v4 hop.

Table 9 isolates the first two discrepancies of Section 5 from the repair made in v4: between v3 and v4 the body of the static proof supplies the same B through the same $-2B$, while the statement halves the coefficient of B , and neither the v3 nor the v4 coefficient is the one that the combination produces, which is $2qB$ inside the root together with a further B outside it. The EUF-CMA term behaves differently: the game hop supplies C once, the published static expression carries $2C$ inside the root in both concrete versions, and the direct combination places C once outside it. There is no analogue of the factor $2q$ for that term. This multiplicity observation is diagnostic rather than conclusive, since a bound may be weakened by enlarging a term; non-implication is established by Proposition 19 and Proposition 24, not here.

This section audits neither the game transitions nor the simulators nor the extraction arguments of the four versions, and makes no claim that they are unsound; the repairs made across the revisions are improvements, taken as given throughout. Our claim is confined to the final algebraic combination which the proofs leave to the reader.

9 Conclusion

In this work we revisited the concrete security bounds of **Sparkle+** in the two settings of [14]: full static security under DL and the EUF-CMA security of DS, and adaptive security for up to $\lfloor t/2 \rfloor$ corrupted parties under AOMDL and the EUF-CMA security of DS, both in the random oracle model. In both, the proofs of [14] display a list of intermediate inequalities and leave their combination to the reader. We carried that combination out, in Theorem 17 and Theorem 22, on the basis of the exact inversion of the general forking lemma isolated in Lemma 14.

The bounds that follow differ from the published expressions in three distinct algebraic respects. First, the terms produced by the two game hops precede the rewinding and, in the direct combination, are added after the GAME 2 success probability has been bounded, so that they appear outside the square root rather than inside it. Second, the extraction inequality reintroduces the whole bad-event aggregate B , respectively B' , of the hop from GAME 1 to GAME 2, and not merely its collision component, inside the root with coefficient $2q$ instead of the coefficient 1 that the published expressions carry; the DS term is not reintroduced by that inequality. Third, the exact inversion of the forking lemma contributes $\frac{q}{2p}$ outside the root together with $\frac{q^2}{4p^2}$ inside it, its standard relaxation combining the two into an additive $\frac{q}{p}$; neither published expression carries any of this. In each of the two settings we exhibit, in Proposition 19 and Proposition 24, values of the quantities involved that satisfy all the displayed inequalities and violate the published expression, so that the latter is not a consequence of the former. The countermodels concern the published expressions as a whole; they do not show that each of the three differences above independently causes the non-implication. Section 8 locates the first appearance of the third difference at the replacement of the local forking lemma by the general one between v2 and v3 of [14], and shows that the first two are independent of the 2025 simulation repair, the static intermediate inequalities being identical in v3 and v4; the adaptive hop, by contrast, did change between those versions, for the documented reason recalled in Remark 25.

At the level of the imported inequalities, what survives of the security claims is the asymptotic conclusion. Conditional on the correctness of the game transitions and reductions from which the intermediate inequalities come, and which this note takes as given rather than auditing, the derived expressions remain negligible under the same assumptions and polynomial resource bounds (Corollary 18 and Corollary 23), so that the qualitative security conclusions for these two corruption settings are unchanged, even though the published concrete expressions are not established by the displayed derivations. Neither of the two expressions is uniformly smaller than the other, and Section 7 gives exact criteria for comparing the published expression with our derived one, in both its exact and its relaxed form, together with a convenient sufficient condition. Under the illustrative model of Section 7.3, when the modelled advantages attain their proxies, and at the instantiation of DS by a Schnorr signature proposed in [14], our derived expression

is numerically smaller than the published one by between twelve and thirty-nine bits at the parameter sets considered.

Three directions seem worth pursuing. The first is to replace the term $n\text{Adv}^{\text{EUF-CMA}}$ by the multi-user unforgeability of DS, as [11] does for its own bound. This would change the reduction underlying the first game hop, and with it the statement of Theorem 16, rather than only its final combination; in the adaptive case a direct analogue would remove only the explicit guessing factor n from $\frac{n^2}{n-t/2} = n \cdot \frac{n}{n-t/2}$, the corruption-survival factor requiring a corruption-aware game or another reduction. Such a replacement could improve the derived estimate, depending on the bound available for the multi-user advantage, which may itself be as large as n times the single-user one. If a tight multi-user reduction did reduce the effective C by a factor n , while leaving the comparison in the regime where the DS term dominates, the modelled gap would widen rather than narrow: there the comparison is $\sqrt{2C}$ against C , and $\log_2(\sqrt{2C}/C) = \frac{1}{2} - \frac{1}{2}\log_2 C$ grows by about $\frac{1}{2}\log_2 n$ as C falls by that factor. The second is to ask whether the $\lfloor t/2 \rfloor$ -adaptive, or the full adaptive, security of Sparkle+ can be proved tightly under a circular-style assumption. The static precedent is [11], and [1] obtains a tight full-adaptive proof of this kind for the original Sparkle rather than for Sparkle+; the point is that the existing AOMDL proof rewinds, which is what produces its square root and what confines its corruption accounting to $\lfloor t/2 \rfloor$, whereas circular-style assumptions have enabled straight-line arguments. The third is to examine whether the concrete bounds of other threshold Schnorr schemes proven by rewinding, FROST [22] among them, place their game-hop terms in the same way.

Acknowledgements

This work grew out of the third author’s studies at the Department of Mathematics, University of Trento; he thanks the other authors for their supervision.

Some claims were made more precise with the assistance of Anthropic Claude and OpenAI ChatGPT, which also reviewed some arguments and proofs. The authors are responsible for all statements and proofs.

References

1. Acharya, O., Fuchsbauer, G., O’Neill, A., Sefranek, M.: Revisiting the security of Sparkle. Cryptology ePrint Archive, Paper 2026/431 (2026), <https://eprint.iacr.org/2026/431>
2. Bacho, R., Chen, Y., Loss, J., Tessaro, S., Zhu, C.: Adaptively secure partially non-interactive threshold Schnorr signatures in the AGM. In: Advances in Cryptology – EUROCRYPT 2026. LNCS, vol. 16541. Springer (2026). https://doi.org/10.1007/978-3-032-25291-3_2, full version: Cryptology ePrint Archive, Paper 2025/1953
3. Bacho, R., Das, S., Loss, J., Ren, L.: Adaptively secure three-round threshold Schnorr signatures from DDH. In: Advances in Cryptology – CRYPTO 2025.

- LNCS, vol. 16005, pp. 390–422. Springer (2025). https://doi.org/10.1007/978-3-032-01887-8_13, full version: Cryptology ePrint Archive, Paper 2025/1009
4. Bacho, R., Loss, J., Tessaro, S., Wagner, B., Zhu, C.: Twinkle: Threshold signatures from DDH with full adaptive security. In: Advances in Cryptology – EUROCRYPT 2024. LNCS, vol. 14651, pp. 429–459. Springer (2024)
5. Baecker, R., Gerhart, P., Li Calsi, D., Russo, L., Schröder, D., Yerukhimovich, A.: Fully adaptive FROST in the algebraic group model from falsifiable assumptions. Cryptology ePrint Archive, Paper 2025/1950 (2025), <https://eprint.iacr.org/2025/1950>
6. Bellare, M., Dai, W., Li, L.: The local forking lemma and its application to deterministic encryption. In: Advances in Cryptology – ASIACRYPT 2019. LNCS, vol. 11923, pp. 607–636. Springer (2019)
7. Bellare, M., Neven, G.: Multi-signatures in the plain public-key model and a general forking lemma. In: Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS). pp. 390–399 (2006). <https://doi.org/10.1145/1180405.1180453>
8. Brandão, L.T.A.N., Peralta, R.: NIST first call for multi-party threshold schemes. Tech. Rep. NIST IR 8214C 2pd (Second Public Draft), National Institute of Standards and Technology, Gaithersburg, MD (Mar 2025). <https://doi.org/10.6028/NIST.IR.8214C.2pd>
9. Canetti, R., Feige, U., Goldreich, O., Naor, M.: Adaptively secure multi-party computation. In: Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC). pp. 639–648 (1996). <https://doi.org/10.1145/237814.238015>
10. Canetti, R., Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Adaptive security for threshold cryptosystems. In: Advances in Cryptology – CRYPTO ’99. LNCS, vol. 1666, pp. 98–116. Springer (1999). https://doi.org/10.1007/3-540-48405-1_7
11. Cho, G., Fuchsbauer, G., O’Neill, A., Sefranek, M.: Schnorr signatures are tightly secure in the ROM under a non-interactive assumption. In: Advances in Cryptology – CRYPTO 2025. LNCS, vol. 16005, pp. 223–255. Springer (2025). https://doi.org/10.1007/978-3-032-01887-8_8, full version: Cryptology ePrint Archive, Paper 2024/1528
12. Ciampi, M., Crites, E., Komlo, C., Maller, M.: On the adaptive security of key-unique threshold signatures. Cryptology ePrint Archive, Paper 2025/943 (2025), <https://eprint.iacr.org/2025/943>
13. Crites, E., Katz, J., Komlo, C., Tessaro, S., Zhu, C.: On the adaptive security of FROST. In: Advances in Cryptology – CRYPTO 2025. LNCS, vol. 16005, pp. 480–511. Springer (2025). https://doi.org/10.1007/978-3-032-01887-8_16
14. Crites, E., Komlo, C., Maller, M.: Fully adaptive Schnorr threshold signatures. Cryptology ePrint Archive, Paper 2023/445, version of 3 June 2025 (2023), <https://eprint.iacr.org/archive/2023/445/1748984094.pdf>
15. Crites, E., Komlo, C., Maller, M.: Fully adaptive Schnorr threshold signatures. Cryptology ePrint Archive, Paper 2023/445, version of 27 March 2023 (2023), <https://eprint.iacr.org/archive/2023/445/1679907858.pdf>
16. Crites, E., Komlo, C., Maller, M.: Fully adaptive Schnorr threshold signatures. Cryptology ePrint Archive, Paper 2023/445, version of 23 June 2023 (2023), <https://eprint.iacr.org/archive/2023/445/1687541087.pdf>
17. Crites, E., Komlo, C., Maller, M.: Fully adaptive Schnorr threshold signatures. Cryptology ePrint Archive, Paper 2023/445, version of 27 May 2024 (2023), <https://eprint.iacr.org/archive/2023/445/1716814228.pdf>

18. Crites, E., Stewart, A.: A plausible attack on the adaptive security of threshold Schnorr signatures. In: *Advances in Cryptology – CRYPTO 2025*. LNCS, vol. 16005, pp. 457–479. Springer (2025). https://doi.org/10.1007/978-3-032-01887-8_15
19. Crites, E.C., Komlo, C., Maller, M.: Fully adaptive Schnorr threshold signatures. In: *Advances in Cryptology – CRYPTO 2023*. LNCS, vol. 14081, pp. 678–709. Springer (2023). https://doi.org/10.1007/978-3-031-38557-5_22
20. Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In: *Advances in Cryptology – CRYPTO ’89*. LNCS, vol. 435, pp. 307–315. Springer (1989). https://doi.org/10.1007/0-387-34805-0_28
21. Gennaro, R., Goldfeder, S.: Fast multiparty threshold ECDSA with fast trustless setup. In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. pp. 1179–1194 (2018). <https://doi.org/10.1145/3243734.3243859>
22. Komlo, C., Goldberg, I.: FROST: Flexible round-optimized Schnorr threshold signatures. In: *Selected Areas in Cryptography (SAC 2020)*. pp. 34–65. LNCS, Springer (2021). https://doi.org/10.1007/978-3-030-81652-0_2
23. Niot, G., Reichle, M., Takemure, K.: Adaptively-secure three-round threshold Schnorr from DL. In: *Advances in Cryptology – EUROCRYPT 2026*. LNCS, vol. 16541. Springer (2026). https://doi.org/10.1007/978-3-032-25291-3_1, full version: Cryptology ePrint Archive, Paper 2025/1941