# Sequential Half-Aggregation of Lattice-Based Signatures

## 1 Introduction

Introduced by [BGLS03], aggregate signature (AS) techniques enable $N$ signers to independently generate signatures $\sigma_1, \ldots, \sigma_N$ on different messages $m_1, \ldots, m_N$, and then aggregate them to produce a single, concise signature $\sigma_{\text{AS}}$. These $\sigma_{\text{AS}}$ can be confirmed according to the verification keys $pk_1, \ldots, pk_N$ of the participants. Certificate chains are among the standard uses of aggregate signatures: Every communication delivered over a public key infrastructure must contain the sender's certificate, which is derived from a chain of certificates that are granted by various authorities. It is crucial to replace the naive concatenation of single-user signatures with a compact, aggregated signature to save bandwidth, as this adds a considerable amount to the certificate chain (e.g., [BGLS03] states that the signature occupies 15% of a typical X.509 certificate length). Two distinct concepts for fully concise aggregate signatures have largely been offered in the literature: (1) Dedicated structures based on bilinear pairings[BGLS03, BNN07], and (2) generic solutions where a signature aggregator yields a concise proof of knowledge of $N$ valid signatures, utilizing iO [HKW15] or non-interactive arguments [DGKV22, WW22, ACL+22].

**Half-Aggregation of Fiat-Shamir Signatures.** Not surprisingly, there aren't many known aggregation techniques designed specifically for Fiat-Shamir signatures [FS87], like Schnorr [Sch91]. [3] Usually, three-round $\Sigma$-protocols are used to produce fiat-Shamir signatures [Cra96]: The signer generates the first-round commit value $u$ by invoking the underlying $\Sigma$-protocol prover, hashes $u$ and the message $m$ to be signed to construct response $z$, and outputs $\sigma = (c, z)$ as a signature. After performing specific algebraic operations to reconstruct $u$ from $(pk, c, z)$, the verifier compares the recomputed hash with $c$. Alternatively, the signer can specify $\sigma = (u, z)$, in which case the verifier recalculates the hash $c$ and determines whether a specific relationship exists between $c$ and $(pk, u, z)$. It is mostly the challenge hash function that makes Fiat-Shamir aggregation difficult: Its usual implementation, SHA-256, lacks an algebraic structure, which makes it incompatible with the pleasant homomorphic characteristics of the fundamental $\Sigma$-protocol transcript. Because of this, the current methods (e.g., [BN06, DEF+19, NRS21]) call for (at least) two phases of interaction before calculating shares of $z$, as each signer must first agree on a combined $u$ that results in the same challenge $c$.

Recent publications [CGKN21, Kas22, CZ22] suggested half-aggregation of Schnorr/EdDSA in order to prevent interaction. These are compromise solutions in which a combination of $N$ partial signatures makes up one part, and only the $u$ or $z$ component is aggregated. While lowering the signature size by a constant factor is asymptotically no better than a naive combination of $N$ signatures, it has practical ramifications; for example, in some scenarios, the complete certificate chain of size $O(N)$ requires communication.

Another strategy that might be used would be to modify one of the generic solutions discussed earlier and have an aggregator node demonstrate that it knows $N$ tuples of the type $(u, c, z)$ that meet the verification requirements described as a circuit. The combination of algebraic operations with non-algebraic hash calculation in verification, however, is probably what determines the prover's complexity. The fact that the security proof of such a general solution would probably rely on heuristics presents another problem when using a common Fiat-Shamir signature. An aggregator of the standard method can only be considered secure if the underlying signature is also guaranteed to be secure, as this needs a specific description of the hash function.

[3] The conversion of interactive multi-signatures to interactive aggregate signatures can be achieved by requesting signatures from all participants on a combination of $N$ messages and public keys, as is widely known [BN06, DEF+19, NRS21]. That being said, this does not fit into the common instances of aggregate

signatures, like a certificate chain, and needs signers to agree in advance on all $N$ messages as well as who they co-sign with.

whereas the bulk of known Fiat-Shamir signatures can only be verified in the randomized oracle model.

**Aggregate Signatures from Lattices.** Considering that two signature contenders, Falcon [PFH+20] and Dilithium [LDK+20], were announced by NIST as part of their post-quantum cryptography standardizing project depending on (structured) lattice assumptions, it is reasonable to wonder if custom aggregate signatures can be generated using lattices rather than generic solutions like [DGKV22, ACL+22].Dilithium adheres to Lyubashevsky's Fiat-Shamir along with Aborts (FSwA) paradigm [Lyu09, Lyu12], whereas Falcon is a GpV-type signature that uses preimage sampleable trapdoor functions [GPV08].

These two finalists reflect the two main design ideas to construct lattice-based signatures. The FSwA paradigm contains a finite number of ideas. A lattice-based implementation of [BN06] was presented by Boneh and Kim [BK20], although it necessitates three interaction rounds. The non-interactive half-aggregation of FSwA has been securely instantiated for the first time by Boudgoust and Roux-Langlois [BR21]. From an abstract standpoint, they modify Schnorr's half-aggregation [CGKN21] to fit the lattice configuration. While the Schnorr algorithm does not really care if we outcome $\sigma = (u, z)$ or $\sigma = (c, z)$, in the lattice context, this distinction is significant. In the second instance, the signature size drastically reduces. All of the u-parts are transmitted during [BR21's half-aggregation, but just the z-parts are aggregated. Keep in mind that transmitting all of the C-parts is insufficient because an aggregated answer is no longer able to retrieve the various commitments. Nevertheless, in order to validate an aggregate signature, we require each and every commitment. Consequently, a signature $\sigma_{\text{AS}} = (u_1, \ldots, u_N, z)$ is produced by the provably secure form of its construction, and it is always greater than the naive combination of $N$ signatures, $\sigma_{\text{con}} = (c_1, z_1, \ldots, c_N, z_N)$. A proposed half-aggregate signature technique known as the MMSAT scheme [DHSS20] is based on the Partial Fourier Recovery problem, an unconventional lattice problem. But it became discovered that even basic forgery attacks exist, and the security proof seems faulty [BR21]. The only lattice-based methods for sequential aggregation that we are aware of adhere to the GPV paradigm [EB14, WW19], the latter of which proves to be unsecure as we illustrate below. In consideration of all of this, we are inspired to pose the following query in this paper:

*Can we create a sequential half-aggregate FSwA signature method that is non-interactive, (1) less in size than the naïve concatenation, and (2) doesn't require the use of pricey generic solutions?*

**Module Lattice Problems**

In addition, we discuss two lattice problems; for further information, see [LS15]. We express them in their discrete, primal, and HNF forms, respectively.

**(M-LWE)**. Assume thatk, $\ell, \eta \in N$. The following is the definition of the Module Learning With Errors problem M-LWE$_{k,\ell,\eta}$. Given $\mathbf{A} \xleftarrow{\$} R^{k\times\ell}$ and $\mathbf{t} \in R_q^k$. Determine if $\mathbf{t} = [\mathbf{A} \mid \mathbf{I}_k] \cdot \mathbf{s}$ or $\mathbf{t} \xleftarrow{\$} R_q^k$, where $\mathbf{s} \xleftarrow{\$} S_\eta^{\ell+k}$.

According to the M-LWE assumption, there isn't a single PPT algorithm that can differentiate between the two distributions with a significant advantage. M-LWE is guaranteed to be not less than as difficult as the approximate shortest vector problem over module lattices, both conventionally [BJRW20] and quantumly [LS15]. This is ensured via reductions from worst-case to average-case scenarios.

**(M-SIS)**. Let $k, \ell, b \in N$. The following is the Module Short Integer Solution problem M-SIS$_{k,\ell,b}$. Given matrix $\mathbf{A} \xleftarrow{\$} R_q^{k\times\ell}$, which is uniformly random. Find a non-zero vector $\mathbf{s} \in R^{k+\ell}$ such that $\|\mathbf{s}\|_2 \leq b$ and $[\mathbf{A} \mid \mathbf{I}_\ell] \cdot \mathbf{s} = \mathbf{0} \in R_q^k$.

No PPT adversary could resolve this problem with significant probability, according to the M-SIS assumption. M-SIS is guaranteed to be classically [LS15] not less than as difficult as the approximate shortest independent vector problem on module lattices by worst-case to average-case reductions.

We now demonstrate Algorithm 2's FH-UF-CMA security (as per Definition 2.9). We additionally outline the security of our plan in Appendix A using a novel model that we refer to as the partial-signature history-free security model, which was presented in [CZ22], for the sake of completeness.

**Theorem 3.4** (**Security** of FH-UF-CMA). *Assume that $n$ is a power of two, $q$ is prime, and Eq. 1 is satisfied for every $k, \ell, n, q, \eta, \gamma, l \in N$. If $s$ is uniformly sampled from $S_\eta^{\ell+k}$ and $\bar{\mathbf{A}} = [\mathbf{A} \mid \mathbf{I}_k]$ with $\mathbf{A}$ uniformly sampled from $R_q^{k\times\ell}$, respectively, then let $p_{inv}$ be the likelihood that $\bar{\boldsymbol{A}}\boldsymbol{s}$ has a minimum of one invertible coefficient over $R_q$. If Algorithm 1's* FSwA-S *signature scheme having message space $M = \{0,1\}^l$ is* UF-CMA *secure, then Algorithm 2's sequential aggregated signature* FSwA-SAS, *or* FH-UF-CMA *secure as well. In specific terms, for every adversary $A$ against* FH-UF-CMA *security that queries the random oracle $H$ at most $Q_h$ times, the OSeqSign oracle at most $Q_s$ times, and produces a forgery having a history of length $N$, there is an adversary $B$ against* UF-CMA *security such that*

$$\mathbf{Adv}^{\text{FH-UF-CMA}}_{\text{FSwA-SAS}}(A) \leq \frac{\mathbf{Adv}^{\text{UF-CMA}}_{\text{FSwA-S}}(B)}{p_{inv}} + O\left(\frac{Q_s(Q_h+Q_s)}{q^{nk/2}}\right) + \frac{(Q_h+Q_s+1)^2}{|Ch|} + \frac{Q_s(2Q_h+1)}{2^l},$$

and $\text{Time}(B) = \text{Time}(A) + O((N+Q_h)k\ell t_{pmul})$, where $t_{pmul}$ is the time of polynomial multiplication in $R_q$.

*Proof.* The high-level ideas of reduction $B$ are first sketched. The entire description of $B$ can be found in Alg 3. For a FH-UF-CMA game (resp. UF-CMA game), $\mathbf{H}$ and also $OSeqSign$ (resp. $\mathbf{H}'$ and $OSign$) stand for the random oracle as well as the signing oracle. After obtaining the challenge public key $t^*$ and the public parameter $\mathbf{A}$, $B$ verifies that $\mathbf{t}^* \in R_q^k$ comprises a minimum of one invertible element. In that case, $B$ transfers ($\mathbf{A}$, $\mathbf{t}^*$) to $A$.

In response to an inquiry, $OSeqSign$ asks $OSign$ to sign a uniformly selected $m$ and programs $\mathbf{H}$ to produce the value $c$ that is returned by an external random oracle $\mathbf{H}'$. Since a forgery filed by $A$ may subsequently utilize the same $m_i$, we are unable to simply pass $m_i$ to $OSign$ in this case. Afterward, $B$ loses the UF-CMA game since their submission of the forgery w.r.t. $m_i$ will be invalid.

The simulation of response to $H$ queries is the core element of reduction. The forgery tuple's key list $LN$ includes $(\mathbf{t}_i, m_i)$ so that $\mathbf{t}_i = \mathbf{t}^*$. Afterward, in order for $(m, (\mathbf{u}_i, \mathbf{z}_i))$ to be considered a legitimate forgery within the UF-CMA game, $B$ had to have extracted the matching $\mathbf{u}_i$ and sent $\mathbf{u}_i$ to $\mathbf{H}'$ as well as a random message $m$. Importantly, $\mathbf{z}_{i-1}$ is utilized in this extraction process whenever $(\tilde{\mathbf{u}}_i, L_i, \mathbf{z}_{i-1})$ is questioned to $\mathbf{H}$. $B$ can extract $\mathbf{u}_i = \tilde{\mathbf{u}}_i - \tilde{\mathbf{u}}_{i-1}$ by using $\mathbf{z}_{i-1}$ as an intuitive look-up key to retrieve the previous aggregated $\tilde{u}_{i-1}$.

To be more precise, we build a number of hybrid games from the initial FH-UF-CMA game in order to arrive at the one that is employed in that final reduction $B$. The probability of which $G_i(A)$ stops with output 1 is indicated by $\Pr[G_i(A)]$.

- $G_0$ (Game 0): This game and the FH-UF-CMA game are identical. First, a blank key-value look-up table HT is initialized by the game. When the table's entry is not empty, a random oracle $\mathbf{H}$ returns HT[X] upon receiving the query with input $X$; If not, it returns $c$, assigns $HT[X] := c$, sample uniform $c \in Ch$. It states that $\Pr[G_0(A)] = \mathbf{Adv}^{\text{FH-UF-CMA}}_{\text{FSwA-SAS}}(A)$.

- $G_1$: This game is precisely the same as $G_0$, with the exception that $OSeqSign$ implements a RO table $\text{HT}[\tilde{\mathbf{u}}_i, L_i, \mathbf{z}_{i-1}] := c_i$ as soon that the rejection sampling phase is successful, and samples uniform $c_i \in Ch$ rather than calling $c_i = \text{H}(\tilde{\mathbf{u}}_i, L_i, \mathbf{z}_{i-1})$ afterwards $\tilde{\mathbf{u}}_i$ is computed. The game terminates by declaring $bad_{ucol} = \text{true}$ if $\text{HT}[\tilde{u}_i, L_i, z_{i-1}]$ has been set. According to this, $|\Pr[G_0(A)] - \Pr[G_1(A)]| \leq \Pr[bad_{ucol}]$.

- $G_2$: With the exception of simulating the answers to random oracle questions $H(\tilde{\mathbf{u}}_i, L_i, \mathbf{z}_{i-1})$, this game appears identical to $G_1$. Create a new, empty ZT key-value lookup table. In the event that $i = 1$ or $X := (\tilde{\mathbf{u}}_{i-1}, L_{i-1}, \mathbf{z}_{i-2})$ exists and $\text{ZT}[X] = \bar{\mathbf{A}}\mathbf{z}_{i-1} \mod q$, extract $\mathbf{u}_i := \tilde{\mathbf{u}}_i - \tilde{\mathbf{u}}_{i-1}$, a sample uniform $c_i \in Ch$, and then assign $\text{ZT}[\tilde{\mathbf{u}}_i, L_i, \mathbf{z}_{i-1}] := u_i + c_i t_i$. In the event that an entry $X' := (\tilde{\mathbf{u}}_{i-1}, L_{i-1}, \mathbf{z}_{i-2})$ already exists and $\text{ZT}[X'] = \mathbf{u}_i + c_i \mathbf{t}_i$, the game terminates by setting $bad_{zcol} = \text{true}$. It holds that $|\Pr[G_1(A)] - \Pr[G_2(A)]| \leq \Pr[bad_{zcol}]$.

- $G_3$: With the exception of $OSeqSign$ and H, which progress as follows, this game resembles to $G_2$. A key-value look-up table MT and an empty set $M$ have been initialized by the game. Upon receiving a query, $OSeqSign$ internally samples the uniform message $m \in M$ then appends $m$ to $M$. After extracting $\mathbf{u}_i$ as previously stated, $H$ sample a uniform message $m \in M$ and aborts by declaring $bad_{mcol} = \text{true}$ if $m \in M$ whenever it receives a query with input $(\tilde{\mathbf{u}}_i, L_i, \mathbf{z}_{i-1})$. It holds that $|\Pr[G_2(A)] - \Pr[G_3(A)]| \leq \Pr[bad_{mcol}]$.

- $G_4$: When the adversary delivers an appropriate signature-history pair $(L_N, (\tilde{\mathbf{u}}_N, \mathbf{z}_1, ..., \mathbf{z}_N))$, this game is the same as $G_3$, with the exception that it validates the ZT entries in the following ways. Let $\tilde{\mathbf{u}}_{N-1}, ..., \tilde{\mathbf{u}}_1$ be the values obtained during the SeqVerify run. The game ends by setting $bad_{ord} = \text{true}$ if for some $i \in [N]$ the entry for $\text{ZT}[\tilde{\mathbf{u}}_i, L_i, \mathbf{z}_{i-1}]$ is undefined. It holds that $|\Pr[G_3(A)] - \Pr[G_4(A)]| \leq \Pr[bad_{ord}]$.

$B$ Assuming that adversary $A$ wins $G_4$, a reduction $B$ mentioned in Alg. 3 can be acquired in this way. After receiving an $OSeqSign$ question, $B$ queries the UF-CMA game's $OSign$ with the uniformity message $m \in M$. $B$ then retrieves $\mathbf{u}_i$ and $\mathbf{z}_i$, programs $HT$ using the challenge $c_i$ returned by the external arbitrary oracle $\text{H}'(\tilde{\mathbf{u}}_i, \mathbf{t}^*, m)$. If H is successful in extracting $\mathbf{u}_i = \tilde{\mathbf{u}}_i - \tilde{\mathbf{u}}_{i-1}$, it also receives new challenge $c_i$ for $\mathbf{t}_i = \mathbf{t}^*$ through querying the outermost random oracle $\text{H}'(\tilde{\mathbf{u}}_i, \mathbf{t}^*, m)$. As $A$ is assured of receiving an invertible challenging public key in $B$, $A$'s perspective is the same as $G_4$'s.

We now demonstrate that, provided that none of the negative events occur, $B$ is guaranteed to provide a message-signature pair $(m, (\mathbf{u}_i^*, \mathbf{z}_i^*))$ that is accepted in the UF-CMA game; that is, $\|\mathbf{z}_i^*\|_\infty \leq B$ and $\mathbf{u}_i^* = \bar{\mathbf{A}}\mathbf{z}_i^* - c_i\mathbf{t}^* \mod q$, where $c_i = \text{H}'(\tilde{\mathbf{u}}_i^*, \mathbf{t}^*, m)$. The former requirement is instantaneous based on $SeqVerify$'s verification condition. To counter the latter, observe that $c_i = \text{H}'(\tilde{\mathbf{u}}_i^*, \mathbf{t}^*, m) = \text{HT}[\tilde{\mathbf{u}}_i^*, L_i^*, \mathbf{z}_{i^*-1}] = c_i^*$ as soon that RO entries $\text{HT}[\tilde{\mathbf{u}}_1, L_1, \mathbf{z}_0], ..., \text{HT}[\tilde{\mathbf{u}}_N, L_N, \mathbf{z}_{N-1}]$ have been set in the correct sequence. As a result, $\mathbf{u}_i^* = \tilde{\mathbf{u}}_i^* - \tilde{\mathbf{u}}_{i^*-1}$ gets extracted through the invocation of $\text{H}(\tilde{\mathbf{u}}_i^*, L_i^*, \mathbf{z}_{i^*-1})$. As it remains that $bad_{zcol} = bad_{ord} = \text{false}$, the following lemma does, in fact, guarantee that such questions have been asked in the correct order.