Overfull: Too Large Aggregate Signatures Based on Lattices

Katharina Boudgoust[1] and Adeline Roux-Langlois[2]
katharina.boudgoust@cs.au.dk, adeline.roux-langlois@irisa.fr

[1] Aarhus University
[2] Univ Rennes, CNRS, IRISA

Abstract. The Fiat-Shamir with Aborts paradigm of Lyubashevsky has given rise to efficient lattice-based signature schemes. One popular implementation is Dilithium which is a finalist in an ongoing standardization process run by the NIST. Informally, it can be seen as a lattice analogue of the well-known discrete-logarithm-based Schnorr signature. An interesting research question is whether it is possible to combine several unrelated signatures, issued from different signing parties on different messages, into one single aggregated signature. Of course, its size should be significantly smaller than the trivial concatenation of all signatures. Ideally, the aggregation can be done offline by a third party, called *public aggregation*. Previous works have shown that it is possible to half-aggregate Schnorr signatures, but it was left unclear if the underlying techniques can be adapted to the lattice setting.
In this work, we show that, indeed, we can use similar strategies to obtain a signature scheme allowing for public aggregation whose hardness is proven assuming the intractability of two well-studied problems on module lattices: The Module Learning With Errors problem (M-LWE) and the Module Short Integer Solution problem (M-SIS).

☹ Failure: Unfortunately, our scheme produces aggregated signatures that are *larger* than the trivial solution of concatenating. This is due to peculiarities that seem inherent to lattice-based cryptography. Its motivation is thus mainly *pedagogical*, as we explain the subtleties when designing lattice-based aggregate signatures that are supported by a proper security proof.

Keywords. Lattice-Based Cryptography, Module Lattices, Signature Aggregation

1 Introduction

For a long time, the main focus of cryptology was on secure encryption. With the invention of public key cryptology in the 1970s and the spread of the internet, the need of secure key exchange and authentication of data became more and more important. This is why nowadays the focus of public key cryptology increasingly shifts towards digital signatures. A digital signature scheme Π_S with message space $\mathcal{M}$ is composed of three algorithms $\mathsf{KGen}, \mathsf{Sig}$ and Vf. The algorithm KGen generates a key pair $(\mathsf{sk}, \mathsf{vk})$ for a given user, who can then use their[3] secret key sk to generate a signature σ on a given message $m \in \mathcal{M}$ by running $\mathsf{Sig}(\mathsf{sk}, m)$. Afterwards, this signature can be verified by anyone using the verification key vk, which is publicly available, by running $\{0,1\} \leftarrow \mathsf{Vf}(\mathsf{vk}, m, \sigma)$. If the verification procedure outputs 1, the signature passes validation.

[3] Throughout the paper, the neutral singular pronouns *they/their* are used in order to keep the language as inclusive as possible. See also `https://www.acm.org/diversity-inclusion/words-matter`

An interesting research question is whether it is possible to define an additional algorithm $\sigma_{agg} \leftarrow \mathsf{AggSig}(\mathsf{VK}, M, \Sigma)$ which takes as input a vector of $N \in \mathbb{Z}$ verification keys $\mathsf{VK} = (\mathsf{vk}_j)_{j\in[N]}$, a vector of N messages $M = (m_j)_{j\in[N]}$ and a vector of N signatures $\Sigma = (\sigma_j)_{j\in[N]}$, that were generated by the N different signing parties with corresponding verification keys vk_j, and outputs a single signature σ_{agg}. We further require a way for others to verify that σ_{agg} is indeed an aggregation of valid signatures. Thus, we need to provide a second additional algorithm $\{0, 1\} \leftarrow \mathsf{AggVf}(\mathsf{VK}, M, \sigma_{agg})$, that outputs 1 if σ_{agg} is a valid aggregation of N valid signatures. All five algorithms define a so-called *aggregate signature scheme* $\Pi_{AS} = (\mathsf{KGen}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{AggSig}, \mathsf{AggVf})$, where we require that it must satisfy correctness and unforgeability properties. A trivial solution is to set σ_{agg} as the concatenation of all the N different signatures and verify one after the other. In the following we are searching for an aggregate scheme that produces a σ_{agg} which is significantly shorter than this trivial solution. Ideally, the aggregation algorithm AggSig can be performed by a third, even untrusted party without needing to communicate with the N signing parties. We call this *public aggregation*. The concept and a first realization of aggregate signatures with public aggregation were given by Boneh et al. [Bon+03] by using bilinear maps constructed over elliptic curves in the generic group model. Aggregate signatures are a useful tool to save communication costs in settings where different users have to authenticate their communication, for instance in consensus protocols or certificate chains. More recently, they attracted increased interest as they help to reduce the size of blockchains such as the Bitcoin blockchain.

Constructing aggregate signature schemes based on the discrete logarithm problem (without bilinear maps) turned out to be much harder, and so far, only solutions that partly aggregate the signatures are known. Chalkias et al. [Cha+21] build a half-aggregate scheme for the well-known Schnorr signature [Sch91]. It produces aggregate signatures of half the size compared to the trivial solution of concatenating. Its security is proven in the Random Oracle Model (ROM) assuming the hardness of the discrete logarithm problem. It was left unclear if the underlying techniques can be adapted to the lattice setting.

Contributions. We propose an aggregate signature allowing public aggregation, whose security is proven assuming simultaneously the hardness of Module Learning With Errors (M-LWE) and Module Short Integer Solution (M-SIS) and thus based on worst-case module lattice problems [LS15]. Earlier proposals either only offered security based on (non-standard) average-case lattice problems, or did not allow for public aggregation (cf. Related Works). From a high level perspective, we take the practical signature from Güneysu et al. [GLP12] as a starting point. It follows the Fiat-Shamir with Aborts (FSwA) paradigm for lattice-based schemes [Lyu12], which is also used in the signature Dilithium [Duc+18], a finalist in the ongoing NIST standardization process[4].

☹ Failure: Due to peculiarities that seem inherent to lattice-based cryptography, our scheme produces aggregated signatures whose sizes are *larger* than the size of the trivial solution (that is concatenating all the single signatures together). The motivation of our work thus is *pedagogical* in order to demonstrate the subtleties when designing lattice-based aggregate signatures that are supported by a proper security proof, in a security model we explain below. We would like to remark that most issues we came across also apply to the MMSA(TK) aggregate signature [Dor+20] (cf. Related Works).

[4] `https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/`

Technical Details. Let us quickly recall the FSwA paradigm for lattice-based signatures in the module setting. In the following, all computations are done over the ring $R_q = \mathbb{Z}_q[x]/\langle x^n + 1\rangle$, where n is a power of two and q is some prime modulus. A verification key is given as $\mathbf{t} = [\mathbf{A}|\mathbf{I}_k]\cdot\mathbf{s} \in R_q^k$, where $\mathbf{s} \in R_q^{\ell+k}$ is a vector of small norm (defining sk), $\mathbf{A} \in R_q^{k\times\ell}$ is a public uniform matrix and $\mathbf{I}_k$ the identity matrix of order k. A signature is provided by $\sigma = (\mathbf{u}, \mathbf{z}) \in R_q^k \times R_q^{\ell+k}$, where $\mathbf{u}$ is a commitment that via some hash function H_c defines a challenge c, and $\mathbf{z}$ is the answer to this challenge. The challenge c is a polynomial with coefficients in $\{-1, 0, 1\}$. For verification, one checks that $\mathbf{z}$ is short and that $[\mathbf{A}|\mathbf{I}_k]\cdot\mathbf{z} = \mathbf{t}\cdot c + \mathbf{u}$, where $c = H_c(\mathbf{u}, \mathbf{t}, m)$ for the verification key $\mathbf{t}$ and a message m. Adding $\mathbf{t}$ to the input of H_c is a simple countermeasure to prevent so-called *rogue-attacks* [Bon+03, Sec. 3.2].

An intuitive way to aggregate N different signatures $(\sigma_j)_{j\in[N]}$ with $\sigma_j = (\mathbf{u}_j, \mathbf{z}_j)$ into one signature σ_{agg} would be to compute the sum of all components $(\sum_j \mathbf{u}_j, \sum_j \mathbf{z}_j)$. However, we would not be able to verify this aggregated signature as we cannot re-compute the different challenges c_j as we do not know the inputs $\mathbf{u}_j$ to H_c, originally used by the signing parties. Thus, we can only sum up the $\mathbf{z}_j$ parts and still have to transmit all the $\mathbf{u}_j$, which produces an aggregate signature of the form $\sigma_{agg} = ((\mathbf{u}_j)_j, \sum_j \mathbf{z}_j)$.[5] This is essentially how our half-aggregate signature looks like. In order to prevent again rogue-type attacks, we use a linear combination (instead of the simple sum), where the coefficients come from some random oracle (which was queried on all signatures that are aggregated). This technique is also used in the Schnorr-analogue by Chalkias et al. [Cha+21]. We formally present our aggregate signature scheme and the rogue-attack for the simple-sum solution in the full version [BR21, Sec. 3+4.3].

☹ Failure: The size of a single signature can be significantly reduced by replacing the commitment $\mathbf{u} \in R_q^k$ by the challenge $c \in R$. This does not only reduce the dimension of the vector from k to 1, but also the total bit-length from $nk\log_2 q$ to $n\log_2 3$, as $\mathbf{u}$ can be any vector in R_q^k but c is a polynomial with ternary coefficients. Unfortunately, this cannot be done in the aggregate setting, as we only have knowledge of the *aggregated* value of all $\mathbf{z}_j$'s. This is the main reason of our failure in constructing aggregate signatures schemes on lattices that are shorter than the trivial concatenation. Note that in the discrete-logarithm setting of the Schnorr aggregate signature in [Cha+21], the challenge c and the commitment $\mathbf{u}$ are elements of the same space. This explains why there is a real improvement for Schnorr signatures, but not for FSwA signatures.

In our aggregate signature, we have to transmit all N vectors $(\mathbf{u}_j)_j$ (and the smallish linear combination of all the $\mathbf{z}_j$), whereas in the trivial concatenation we transmit all N challenges $(c_j)_j$ (and the N small vectors $\mathbf{z}_j$). The size of the $\mathbf{u}_j$ is so large that it annihilates the compressing effect of combining all the $\mathbf{z}_j$. More concretely, taking the level III parameters of Dilithium [Duc+18] and $N = 1000$ signatures to aggregate, then $(\mathbf{u}_1, \ldots, \mathbf{u}_N) \in (R_q^k)^N$ is of size ca. 4400 KB.[6] However, simply concatenating N single Dilithium signatures produces an aggregate signature of a smaller size of ca. 3300 KB. Note that in an earlier version of [BR21] we further compressed the aggregated signature via some linear function T to obtain a solution that was indeed smaller than the trivial concatenation. As we elaborate in [BR21, Sec. 3.3], this allowed for simple lattice attacks.

[5] Chalkias et al. [Cha+21, Sec. 6] provide evidence that it is *necessary* to transmit all the commitments.

[6] This calculation does not even take into account the fact that in our aggregate signature $\log_2 q$ has to be increased by some factor $\log_2 \sqrt{N}$.

In the full version, we provide a rigorous security proof (Theorem 1), where the proof idea follows the one of Damgård et al. [Dam+21] for their inter-active multi-signature (cf. Related Works). It is composed of a sequence of indistinguishable games (assuming the hardness of M-LWE), where the starting one is the security game of our aggregate signature. The game is specified by the aggregate chosen key model, as introduced by Boneh et al. [Bon+03]. In the last game, the signing procedure is simulated by some algorithm that doesn't depend on the secret key and the verification key is sampled uniformly at random. By applying (twice) the General Forking Lemma from Bellare and Neven [BN06] we can use four different responses of a successful adversary against the scheme in the last game to solve an instance of M-SIS. As we don't need trapdoor commitment schemes, the proof is less technical than the one in [Dam+21]. We use a Gaussian distribution for the masking vectors, the rejection sampling and the linear combination as done in [Dam+21]. This leads to tighter norm bounds of an aggregate signature.

Related Works. A first attempt to build lattice-based aggregate signatures with public aggregation was made by Doröz et al. [Dor+20]. Their construction builds upon the signature scheme PASS Sign, introduced by Hoffstein et al. [Hof+14]. As a warm-up, they introduce a simple linear half-aggregate signature, which they call MMSA (multi-message, multi-user signature aggregation). However, in this version, the aggregate signature is larger than the trivial concatenation of N different signatures. In order to improve the efficiency, they first compress the signature, leading to MMSAT (the T stands for a linear compression function T), and then compress the verification keys, leading to MMSATK. Unfortunately, their construction has several disadvantages: First, the linear compression used in MMSAT(K) is prone to simple forgery attacks (similar to [BR21, Sec. 3.3]), making those constructions insecure. Second, they only provide a security proof for the first variant MMSA by showing that it inherits the security of the underlying PASS Sign, and subsequently its security can be based on the hardness of the Partial Fourier Recovery problem (PFR). The PFR asks to recover a polynomial in the ring $\mathbb{Z}[x]/\langle x^n - 1\rangle$ of small norm having access only to a partial list of its Fourier transform. It can be formulated as a bounded distance decoding problem over some ideal lattice. This may rise security concerns, as problems over ideal lattices have been shown to be in specific parameter settings easier than their counterparts over arbitrary lattices, e.g., [CDW21]. Furthermore, up to today, there are no connections to worst-case lattice problems, which may be seen as an additional security concern.

In a parallel line of research, aggregate signature schemes that only allow for *private aggregation* have been proposed. In this setting, the different signing parties interact with each other to generate an aggregated signature on one message, which can be the concatenation of different messages. Those are also known as *multi-signature schemes* and there have been several recent protocols following the FSwA paradigm providing lattice-based inter-active aggregate signatures, see for instance [Dam+21] and references therein.

Open Problem: We leave as an open problem the construction of an aggregate signature scheme based on standard lattice-problems which allows public aggregation, produces aggregate signatures that are smaller than the simple concatenation and at the same time is provably secure in the aggregate chosen-key security model.

Acknowledgments

This work has received a French government support managed by the National Research Agency under the projet AMIRAL with reference ANR-21-ASTR-0016. It was also funded by the Danish Independent Research Council under project number 0165-00107B (C3PO).

We warmly thank Akira Takahashi for making us aware of the attacks against Schnorr-based aggregate signatures in the chosen-key model and discussions with him and Claudio Orlandi. We also would like to thank Olivier Sanders for an interesting exchange regarding the security proof. Lastly, we also would like to thank Thomas Prest who explained us his attack on the linear compression function used in an earlier version of this paper.

References

[BN06] Mihir Bellare and Gregory Neven. “Multi-signatures in the plain public-Key model and a general forking lemma”. In: *CCS*. ACM, 2006, pp. 390–399.

[Bon+03] Dan Boneh et al. “Aggregate and Verifiably Encrypted Signatures from Bilinear Maps”. In: *EUROCRYPT*. Vol. 2656. Lecture Notes in Computer Science. Springer, 2003, pp. 416–432.

[BR21] Katharina Boudgoust and Adeline Roux-Langlois. “Compressed Linear Aggregate Signatures Based on Module Lattices”. In: *IACR Cryptol. ePrint Arch.* 2021 (2021), p. 263. URL: `https://eprint.iacr.org/2021/263`.

[CDW21] Ronald Cramer, Léo Ducas, and Benjamin Wesolowski. “Mildly Short Vectors in Cyclotomic Ideal Lattices in Quantum Polynomial Time”. In: *J. ACM* 68.2 (2021), 8:1–8:26.

[Cha+21] Konstantinos Chalkias et al. “Non-interactive Half-Aggregation of EdDSA and Variants of Schnorr Signatures”. In: *CT-RSA*. Vol. 12704. Lecture Notes in Computer Science. Springer, 2021, pp. 577–608.

[Dam+21] Ivan Damgård et al. “Two-Round n-out-of-n and Multi-signatures and Trapdoor Commitment from Lattices”. In: *Public Key Cryptography (1)*. Vol. 12710. Lecture Notes in Computer Science. Springer, 2021, pp. 99–130.

[Dor+20] Yarkin Doröz et al. “MMSAT: A Scheme for Multimessage Multiuser Signature Aggregation”. In: *IACR Cryptol. ePrint Arch.* (2020), p. 520.

[Duc+18] Léo Ducas et al. “CRYSTALS-Dilithium: A Lattice-Based Digital Signature Scheme”. In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2018.1 (2018), pp. 238–268.

[GLP12] Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann. “Practical Lattice-Based Cryptography: A Signature Scheme for Embedded Systems”. In: *CHES*. Vol. 7428. Lecture Notes in Computer Science. Springer, 2012, pp. 530–547.

[Hof+14] Jeffrey Hoffstein et al. “Practical Signatures from the Partial Fourier Recovery Problem”. In: *ACNS*. Vol. 8479. Lecture Notes in Computer Science. Springer, 2014, pp. 476–493.

[LS15] Adeline Langlois and Damien Stehlé. “Worst-case to average-case reductions for module lattices”. In: *Des. Codes Cryptogr.* 75.3 (2015), pp. 565–599.

[Lyu12] Vadim Lyubashevsky. “Lattice Signatures without Trapdoors”. In: *EUROCRYPT*. Vol. 7237. Lecture Notes in Computer Science. Springer, 2012, pp. 738–755.

[Sch91] Claus-Peter Schnorr. “Efficient Signature Generation by Smart Cards”. In: *J. Cryptol.* 4.3 (1991), pp. 161–174.