How SOLMAE was designed

Kwangjo Kim

IACR Fellow, Emeritus Professor of KAIST,
President of International Research Institute for Cyber Security(IRCS), Korea
kkj@kaist.ac.kr

Abstract. This paper briefly introduces the design rationale and its reference implementation of SOLMAE which is a lattice-based signature scheme following the hash-and-sign paradigm in the style of Gentry–Peikert–Vaikuntanathan signatures, and instantiated over NTRU lattices. SOLMAE shows the same simple, fast, parallelizable signing algorithm as Mitaka, with flexible parameters by leveraging a novel key generation algorithm that is much faster and achieves versatile security and short key and signature sizes as FALCON. In addition, SOLMAE is strongly believed to be one of best KpqC candidates as lattice-based signature over structured lattices by judging its reference implementation compared with others.

Keywords: Signature schemes · Lattice-based cryptography · Hash-and-sign paradigm · Parallel key generation · Lattice Gaussian sampling

1 Introduction

When Shor [10] has proposed an efficient randomized algorithm on a hypothetical quantum computer in 1999 to integer factorization and discrete logarithm problems in a polynomial time, it was beyond of our imagination building for the powerful computing environment at that time. Currently the threat of attacking the current (or classical) secure system by using the quantum computer is expected to be right at our fingertips due to the aggressive road map by IBM quantum computing. We are very concerned about so called *Harvest now, decrypt later* attack [11] which is a surveillance strategy that relies on the acquisition and long-term storage of currently unreadable encrypted data awaiting possible breakthroughs in decryption technology that would render it readable in the future.

Due to the substantial amount of research on quantum computers, large-scale quantum computer if built, can break many of public-key cryptosystems in use. In 2016, NIST [9] has initiated Post Quantum Crytography(PQC) project to solicit, evaluate, and standardize one or more quantum-resistant cryptographic algorithms for Key Encapsulation Mechanism(KEM) and Digital Signature(DS) from all over the world. After several number of rounds, NIST has finally selected CRYSTALS-Kyber for KEM and CRYSTALS-Dilithium, FALCON and SPHINCS+ in 2022.

Influenced by this NIST PQC project, Korean cryptographic society led by KpqC organization [8] has called for soliciting Korean PQC standard candidates by the end of Oct. in 2022. By the due of submission, 7 candidate algorithms and 8 candidate algorithms for KpqC competition were submitted for KEM and DS, respectively.

SOLMAE [1] submitted to KpqC Competition as one of DS candidate algorithms is a lattice-based signature scheme inspired by several pioneering works based on the hash-then-sign signature paradigm proposed by Gentry, Peikert and Vaikuntanathan [5]. In this paper, we report how SOLMAE was designed depending on design rationale and suggest its performance compared with FALCON and ECDSA using our reference implementation in C programming language as Proof-of-Concept.

The organization of this paper is as follows: In **Section 2**, we describe the design rationale of SOLMAE and suggest its pros and cons in **Section 3**. In **Section 4**, we make comparison SOLMAE with RSA, ECDSA and FALCON from the points of security and performance. Finally we will give the concluding remarks and challenging issues.

2 Design rationale

SOLMAE is inspired from FALCON's design. Some of the new theoretical foundations of our

[1] This is an acronym of quantum–**S**ecure alg**O**rithm for **L**ong–term **M**essage **A**uthentication and **E**ncryption.

scheme were laid out in the presentation of Mitaka [3]. At a high-level, it removes the inherent technicality of the sampling procedure, and most of its induced complexity from an implementation standpoint, for *free*, that is with no loss of efficiency. The simplicity of our design translates into faster operations while preserving signatures and verification keys sizes, on top of allowing for additional features absent from FALCON, such as enjoying cheaper masking, and being parallelizable. By using the novel compression techniques and tools of [4], we can also obtain smaller signatures and verification keys than those already achieved by FALCON. To sum-up, our design aims to achieve *better performances* for the same security and advantages as FALCON.

Here we focus on the big lines behind our scheme's principles, keeping details at a minimum. While its predecessor FALCON could be summed-up as *an efficient instantiation of the GPV framework*, SOLMAE takes it one step further. The main ingredients in SOLMAE are:

- **Hybrid sampler** is a faster, simpler, parallelizable and maskable Gaussian sampler to generate signatures;
- **Optimally tuned key generation algorithm**, enhancing the security of our new sampler to that of FALCON's level[2];
- **Dedicated compression techniques** to reduce bandwidth consumption even further, at no cost on the security according to our analyses.

A quick overview of hash-then-sign over lattices Almost all hard cryptographic problems from lattices involve either computing short vectors or decoding a target to a close lattice point, from an arbitrarily bad description of the lattice. Hash-then-sign over lattices is no exception, as it can be described as follows:

- a message M is hashed as a vector $m = H(M)$ in the ambient space of a *public* lattice $\mathcal{L}$;
- After computing a point $v \in \mathcal{L}$ *quite close* to $H(M)$, a signature is $s = H(M) - v$;
- a pair (M, s) is valid if $H(M) - s$ belongs to $\mathcal{L}$ and s is short enough.

On the one hand, only the signer should be able to *efficiently* compute v close enough to an arbitrary target. This is a decoding problem that can be solved when a basis of *short* vectors is known. On the other hand, anyone wanting to check the validity of a signature should be able to verify lattice membership. The scheme therefore relies on two main ingredients:

1. the ability to generate trapdoor pairs $(\mathbf{A}, \mathbf{B})$ of bases for a given lattice $\mathcal{L}$, where $\mathbf{B}$ remains secret and is composed of short vectors;
2. an efficient procedure exploiting trapdoor, $\mathbf{B}$ to compute signatures.

Gentry-Peikert-Vaikuntanathan paradigm The key observation in the GPV framework [5] is that one can get a non-leaking signature algorithm by replacing the round-off by *lattice Gaussian sampling*. Such a procedure would output random vectors in $\mathcal{L}_{\text{NTRU}}$ with a Gaussian-like distribution independent of the lattice basis, thwarting statistical attacks to recover $\mathbf{B}$. The authors of [5] also recalled Klein's algorithm to sample lattice Gaussians in quadratic time, but the *practical* efficiency of the whole design was not really addressed. This result was undeniably a huge step forward for hash-then-sign over lattices, however trapdoors were now not only asked to be made of short vectors, they would also need that their *Gram-Schmidt orthogonalization* was made of vectors as short as possible. This was indeed necessary to ensure that Klein's algorithm would output Gaussians with small variance; without such a property, it would be easier for an adversary to forge valid signatures.

NTRU strikes again: the trapdoors of Ducas, Lyubashevsky and Prest In [1], NTRU-like lattices were again the center of attention. Switching from $\mathbb{Z}[X]/(X^n - 1)$ to a ring of cyclotomic integers $R = \mathbb{Z}[X]/(X^{2^n} + 1)$, the authors observed that the algebraic structure underlying these lattices gave strong and useful geometric constraints. Generally, the largest Gram-Schmidt vectors of an NTRU lattices would correspond to (f, g) and a completion (F, G) of the secret basis. It is hopeless to expect *any* (f, g) to lead to a trapdoor basis useful for signing, but one could hope to find a good pair reasonably fast by some kind of random walk among the set of potential keys.

Sampling: an interplay between trapdoor quality and security level Klein's algorithm actually suffers from its quadratic complexity in practice. Because of the algorithm's design, it also unfriendly to parallelization. Ducas and Prest

[2] This corresponds to NIST-I and NIST-V requirements.

soon realized that these limitations could be avoided thanks to the algebraic structure of the underlying cyclotomic ring $\mathbb{Z}[X]/(X^{2^n}+1)$. They described a recursive *quasi-linear* approach [2] to Klein's algorithm exploiting the *tower* structure of the ring, in the spirit of the Fast Fourier Transform algorithm — which gave its name to their new sampler. At the price of an intricate implementation, the resulting Gaussian sampler achieves impressive performances, close to signing time of the other NIST winning signature, DILITHIUM.

3 Pros and Cons

The readers can refer the details of SOLMAE to [7]. SOLMAE enjoys the following pros.

- **Compactness:** The signature size, or the combined verification key plus signature size, are comparable to that of FALCON's, which was the lightest in bandwith consumption among the winning signatures in NIST's competition. They can be further reduced by the addition of the compression techniques of [4].
- **Simplicity and efficiency:** The hybrid sampler is tailored to exploit the algebraic structures of NTRU lattices, involves only straightforward, elementary operations between polynomials, and is practically more efficient than the FFO sampler.
- **Side-channel resilience:** Masking SOLMAE can be done with standard and well-understood counter-measures, at cheaper overhead than FALCON.

On the other hand, our scheme presents some cons:

- **Reliance on floating point arithmetic:** Just like its ancestor FALCON, our scheme relies importantly on the Fourier representation of polynomials, that is, representing them by evaluation at complex roots of unity, prompting the use for floating points arithmetic.
- **Algebraically structured security assumptions:** NTRU lattices enjoys strong symmetries stemming from their algebraic structure, meaning that the underlying hardness assumption corresponds to a subclass of problems potentially easier than for plain, regular lattices. We stress that up to current knowledge, no significant improvement on the cryptanalysis or the asymptotic complexity are known for these problems.

4 Comparison and Performance Analysis

Table 1 summarizes the comparison of key features between RSA and SOLMAE from the points of security and computation. Note that we need algebraic knowledge to understand SOLMAE and SIS means Short Integer Solution.

Table 1: Key Features of RSA and SOLMAE

Item	RSA	SOLMAE
Mathematics	Number Theory	Algebra
Basic operation	Mod. Exp.	Polynomial
Trapdoor	Mul. Inverse	NTRU
Verification	$Left = Right$	$Left \leq Right$
Gaussian sampling	No	Yes
Security assumption	Integer Fact.	SIS
Worst to avg. red.	No	Yes
Classical attack	No	No
Quantum attack	Yes	No

Our implementation has been tested on various x86–64 platforms, and consistently outperforms FALCON in signing and verification in equal dimension, while key generation is slightly slower. Timings below have been collected on a single core of a Ryzen Threadripper Pro 5975WX @ 3.60 GHz workstation with hyperthreading and frequency scaling disabled. The performance comparison between SOLMAE and FALCON is made in Table 2 where since the `speed` tool used in FALCON does not include cycle counts, so they are omitted. In Table 2, S-512(or 1024) indicates SOLMAE-512(or 1024) and F-512(or 1024) indicates FALCON-512(or 1024), respectively.

Table 2: Comparison between SOLMAE and FALCON

		S-512	S-1024	F-512	F-1024
`KeyGen` time	Mcycles	27	65	—	—
	time (ms)	7.5	18	5.0	15
`pk` size	Bytes	896	1792	896	1792
`Sign` time	Kcycles	387	775	—	—
	time (μs)	108	216	220	441
`sgn` size	Bytes	666	1375	666	1280
`Verif` time	Kcycles	40	84	—	—
	time (μs)	11	23	18	36

Observe that our new `KeyGen` is fairly close to FALCON's in terms of speed. For signing, SOL-

MAE consistently outperforms FALCON by a factor of about 2, and for verification, SOLMAE is also about 50% faster.

In Table 3 we made performance comparison of SOLMAE-512 with Shake-128 and ECDSA P256r1 with SHA256 provided by Dreamsecurity Engineer [6] using our publicly available SOLMAE-512 C language reference implementation under their typical computing platform. Compared with ECDSA, the keygen of SOLMAE-512 is slower than EDCSA P256r1 by online computation [3]. But the signing and verification of SOLMAE-512 is about 10 times more faster than those of ECDSA P256r1 currently used in TSL or SSL which makes no speed degradation when we apply quantum-secure SOLMAE for PKI and various security applications.

Table 3: Comparison of SOLMAE and ECDSA

		SOLMAE	ECDSA
Specification		512	P256r1
Size(Bytes)	`pk`	1,792	65
	`sgn`	1,375	32
Time	`KeyGen`(ms)	30.21	2.53
	`Sign`(μs)	288.2	2,582.8
	`Verif`(μs)	55.6	7,744.7

5 Concluding Remarks

FALCON is claimed to have the advantage of providing short public keys and signatures as well as high security levels; plagued by a contrived signing algorithm, not very fast for signing and hard to parallelize; very little flexibility in terms of parameter settings.

However, our design shows that SOLMAE has the simple, fast, parallelizable signing algorithm, with flexible parameters. By leveraging a novel key generation algorithm that is much faster and achieves higher security, SOLMAE achieves the versatile security and short key and signature sizes as FALCON. From the performance point of view, SOLMAE was verified to be faster than ECDSA when signing and verifying and is strongly believed to be one of best candidates for KpqC competition. Some challenges such as implementation of intermediate NIST security level from II to IV and the extension of SOLMAE for various applications *etc.* are left to do next.

[3] Note that keygen of SOLMAE can be operated online/offline together in parallel.

Acknowledgement

This work was partially supported by Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean Government (20ZR1300, Core Technology Research on Trust Data Connectome).

References

1. Ducas, L., Lyubashevsky, V., Prest, T.: Efficient identity-based encryption over NTRU lattices. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014, Part II. LNCS, vol. 8874, pp. 22–41. Springer, Heidelberg (Dec 2014). https://doi.org/10.1007/978-3-662-45608-8_2 2
2. Ducas, L., Prest, T.: Fast fourier orthogonalization. In: Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation, ISSAC 2016, Waterloo, ON, Canada, July 19-22, 2016. pp. 191–198. ACM (2016) 3
3. Espitau, T., Fouque, P.A., Gérard, F., Rossi, M., Takahashi, A., Tibouchi, M., Wallet, A., Yu, Y.: Mitaka: a simpler, parallelizable, maskable variant of falcon. Advances in Cryptology, Proc. of EUROCRYPTO 2022, Part III pp. 222–253 (2022) 2
4. Espitau, T., Tibouchi, M., Wallet, A., Yu, Y.: Shorter hash-and-sign lattice-based signatures. Advances in Cryptology, Proc. of CRYPTO 2022, Part II pp. 245–275 (2022) 2, 3
5. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Ladner, R.E., Dwork, C. (eds.) 40th ACM STOC. pp. 197–206. ACM Press (May 2008). https://doi.org/10.1145/1374376.1374407 1, 2
6. Kim, H.: Personal correspondence (2023), will provide upon request 4
7. Kim, K., Tibouchi, M., Espitau, T., Takashima, A., Wallet, A., Yu, Y., Guilley, S., Kim, S.: Solmae : Algorithm specification. Updated SOLMAE, IRCS Blog (2023), https://ircs.re.kr/?p=1714 3
8. KpqC: Korean post-quantum crytography (2020), https://kpqc.or.kr/ 1
9. NIST: Post-quantum crytography (2016), https://csrc.nist.gov/projects/post-quantum-cryptography 1
10. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Review **41**(2), 303–332 (1999) 1
11. Wikipedia: Harvest now, decrypt later (2023), https://en.wikipedia.org/wiki/Harvest_now_decrypt_later 1