Accelerating Bliss: the geometry of ternary polynomials

Léo Ducas

University of California, San Diego
lducas@eng.ucsd.edu

Abstract

The signature scheme Bliss proposed by Ducas, Durmus, Lepoint and Lyubashevsky at Crypto'13, is currently the most compact and efficient lattice-based signature scheme that is provably secure under lattice assumptions. It does compare favourably with the standardized schemes RSA and ECDSA on both Software and Hardware.

In this work, we introduce a new technique that improves the above scheme, offering an acceleration factor up to 2.8, depending on the set of parameters.

Namely, we improve the *unnatural* geometric bound used in Bliss to a tighter and much more natural bound by using some extra degree of freedom: the ternary representations of binary challenges. Precisely, we efficiently choose a ternary representation that makes the result *deterministically shorter* than the *expected* length for a random challenges.

Our modified scheme Bliss-b is rather close to the original scheme, and both versions are compatible. The patch has been implemented on the Open-Source Software implementation of Bliss, and will be released under similar license.

1 Introduction

Lattice based cryptography [Ajt96, AD97], has received a lot of attention in the last decade, mostly for theoretical interest. First for security: lattices enjoy worst-case hardness properties and seem resistant to quantum attacks. Later, lattices have shown exceptional versatility, allowing new constructions such as the breakthrough result of Gentry [Gen09] for Fully Homomorphic Encryption.

On the practical side, lattice-based cryptography has long been a *promising* alternative to RSA and discrete-log cryptography. Indeed, the use of *algebraic lattices* [Mic02] can boost many lattice-based constructions as already noticed in the late 90's for the construction of the NTRU Encryption scheme[1] [HPS98]. While efficient lattice-based encryption has quickly found satisfactory solutions, lattice-based signatures have been more problematic: the natural lattice trapdoor leaks some secret information [NR06, DN12]. It is only in 2008 that provably secure solutions to prevent such leakage have been found [GPV08, Lyu09].

Both solutions were originally not so practical, and could only be proved secure in the random-oracle model. Despite several efforts [SS11, DPL14] the *hash-then-sign* approach of [GPV08] is indeed quite slow for signatures in practice. But the trapdoors of [GPV08, MP12] remain useful for more advanced constructions such as attribute based encryption [CHKP12, GVW13, DPL14] or to avoid relying on random oracles [Boy10, DM14] when building signatures.

[1] IEEE Std 1363.1 and standard X9.98.

On the other hand, the *Fiat-Shamir with aborts* approach of Lyubashevsky [Lyu09, Lyu12, GLP12, DDLL13, BG14, PDG14, MBDG14] has demonstrated very good performances. In particular, the recent scheme Bliss has proved to compete with the standardized signature schemes RSA and ECDSA both on software and hardware [DDLL13, PDG14]. Nevertheless, Bliss seems to have room for improvements, both at the implementation level[2] and at the design level.

In particular, there is a geometric constant C used to bound the length of certain vectors during the execution of Bliss. Because of the abort technique, this constant C has a direct impact on the overall speed of the protocol. Experiments suggest that this bound should not be much larger that 1: over 20000 random challenges, $C = 1.1$ is sufficient with some margin. Yet we need this bound to hold for all keys and all challenges, for which Ducas *et al.* [DDLL13] could only prove, a constant $C \in [1.5, 1.9]$ depending on the parameter set.

1.1 This work

The problem. The speed of Bliss is related to the ℓ_2 norm of the product $\mathbf{Sc}$ where the matrix $\mathbf{S}$ is the secret key and $\mathbf{c}$ the challenge vector. More precisely, one requires a bound $\|\mathbf{Sc}\| \leq B$, independent of the secret key $\mathbf{S} \in \mathcal{S}$ in order to avoid leaking information on the secret key $\mathbf{S}$ while signing. This bound dictates the running time of the Bliss signing procedure: the main loop needs to be repeated $M = e^{B^2/2\sigma^2}$ times on average[3], and M is called the repetition rate. Therefore any improvement on this bound $\|\mathbf{Sc}\| \leq B$ directly leads to an acceleration of Bliss without altering the rest of the scheme.

The set of challenges $\mathbb{B}_\kappa^n$ is the set of binary vector with exactly κ entries equal to 1, in particular we have $\|\mathbf{c}\| = \sqrt{\kappa}$. Quite naturally one could bound $\|\mathbf{Sc}\|$ by $s_1(\mathbf{S}) \cdot \|\mathbf{c}\|$ where $s_1(\mathbf{S})$ denotes the singular norm (a.k.a. the spectral radius) of $\mathbf{S}$. Unfortunately the singular norm of $\mathbf{S}$ is quite larger in the ring setting than in the standard setting: for ternary entries $\{0, \pm 1\}$ with density δ of ± 1 entries, we expect $s_1(\mathbf{S}) \approx \sqrt{\delta n}\sqrt{\log n}$ in the ring setting against $s_1(\mathbf{S}) = \Theta(\sqrt{\delta n})$ without ring structure (see Section 2.3). This $\sqrt{\log n}$ factor may seem anecdotal in theory but is quite relevant in practice: in the design of Bliss a lot of effort was done to cut such corners, pushing the proof of concepts [Lyu09, Lyu12] to an actual, practical scheme, competing with standardized cryptography [DDLL13, PDG14].

To improve on such a bound, Ducas *et al.* [DDLL13] exploit the binary structure of the challenge vector $\mathbf{c}$ and discuss the Gram matrix of the secret matrix $\mathbf{G} = \mathbf{S}^t\mathbf{S}$. In practice, rejecting a reasonable (up to 90%) proportion of secret keys $\mathbf{S}$ during key generation depending on $\mathbf{G}$ they obtain the provable guarantee that $\|\mathbf{Sc}\| \leq C\delta\sqrt{n\kappa}$ for any $\mathbf{c} \in \mathbb{B}_\kappa^n$, where the value C varies between 1.5 and 1.9 from Bliss-0 to Bliss-IV. While this is much better than the singular norm bound (in practice measured to be around 5), this is still quite far from 1.

Our solution. The introduction of the bimodal trick in [DDLL13] had a small drawback compared to previous similar constructions [Lyu09, Lyu12, GLP12]: the challenge vector $\mathbf{c}$ inherently became a modulo 2 object, and it was no longer possible to increase the entropy by using coefficients in $\{0, \pm 1\}$ rather than $\{0, 1\}$.

Nevertheless, it does not restrict us to using the canonical binary representation $\mathbf{c}' \in \mathbb{Z}^n$ of $\mathbf{c} \in \mathbb{Z}_2^n$, in particular one may negate at will individual coordinates of $\mathbf{c}'$ at the beginning of the

[2]The open source software implementation [DL] does not use vectorized operations (MMX/SSE) for the Number Theoretic Transform. Pseudo random number generation could also benefit from AES instructions.

[3]the choice of σ is determined by security constraints, which we won't discuss in this paper.

response phase, as long as the bound on $\|\mathbf{Sc}'\|$ is preserved. And in fact, choosing an appropriate representation $\mathbf{c}'$ may give a smaller result. We propose a simple algorithm that efficiently compute the product $\mathbf{v} = \mathbf{Sc}'$ for some $\mathbf{c}' \equiv \mathbf{c} \bmod 2$ such that $\|\mathbf{Sc}'\| \leq \|\mathbf{S}\| \cdot \sqrt{\kappa}$ (where $\|\mathbf{S}\|$ denote the largest norm among the columns of $\mathbf{S}$). This result is optimal considering the case where $\mathbf{S}$ would have perfectly orthogonal rows.

Results. Using the technique above, we are able to remove the unnatural constant $C \in [1.5, 1.9]$ and replace it by 1. We propose a variant Bliss-b of Bliss, with similar set of parameters but for the rejection rate M, which is improved by a factor from 1.4 to 3.4. We have implemented our modification on the open-source implementation [DL]. Due to additional algorithmic efforts, the speed-up in practice is slightly less than what is predicted by the rejection rate. Our modification also speeds-up the key generation by a factor 5 to 10. Our patch will be made available under the terms of the original CeCILL license.

Organization. The rest of the paper is organized as follows. In Section 2 we give the necessary preliminaries on the construction of Bliss. Then, in Section 3 we present the efficient sign choosing algorithm and prove the associated bound on the product $\|\mathbf{Sc}'\|$. We conclude with Section 4 by detailing the modification to obtain the new scheme Bliss-b: the algorithmic modifications, the new parameters, and the comparative benchmarks.

2 Preliminaries

Notations Lower-case bold letters will denote column vectors over $\mathbb{Z}$, and upper-case bold letters will denote matrices. The norm $\|\cdot\|$ notation refers to the euclidean norm for vectors. For a matrix $\mathbf{M} = [\mathbf{m}_1 \dots \mathbf{m}_k] \in \mathbb{Z}^{\ell \times k}$, the norm $\|\mathbf{M}\|$ refers to the maximal norm among its columns: $\|\mathbf{M}\| = \max_i \|\mathbf{m}_i\|$. We note $\mathbb{B}$ for the binary set $\{0, 1\}$, and $\mathbb{B}_\kappa^n$ denotes the set of vectors of n binary coordinates with exactly κ coordinates set to 1: $\mathbb{B}_\kappa^n = \{\mathbf{c} \in \mathbb{B}^n | \|\mathbf{c}\|_1 = \kappa\}$. For such binary vectors $\mathbf{c} \in \mathbb{B}_\kappa^n$, we define the indicator set $\mathcal{I}_\mathbf{c} = \{i | \mathbf{c}_i \neq 0\}$.

2.1 The Cyclotomic Ring

As several prior constructions [LMPR08, LPR10], Bliss relies on a power of 2 cyclotomic ring: we let n be a power of 2 defining the $(2n)$th cyclotomic polynomial $\Phi_{2n}(X) = X^n + 1$ and associated cyclotomic ring $\mathcal{R} = \mathbb{Z}[X]/(X^n + 1)$. Those rings have convenient geometric properties, and allows very efficient Number Theoretic Transform for well chosen modulus q. We also write $\mathcal{R}_q = \mathcal{R}/(q\mathcal{R})$ for the residue ring of $\mathcal{R}$ modulo an integer q. Elements in $\mathcal{R}$ have a natural representation as polynomials of degree $n - 1$ with coefficients in $\mathbb{Z}$, and $\mathcal{R}$ can be identified (as an additive group) with the integer lattice $\mathbb{Z}^n$, where each ring element $a = a_0 + a_1 x + \ldots + a_{n-1} x^{n-1} \in \mathcal{R}$ is associated with the coefficient vector $\mathbf{a} = (a_0, \ldots, a_{n-1}) \in \mathbb{Z}^n$.

The ring $\mathcal{R}$ is also identified with the sub-ring of anti-circulant square matrices of dimension n by regarding each ring element $r \in \mathcal{R}$ as a linear transformation $x \mapsto r \cdot x$ over (the coefficient embedding) of $\mathcal{R}$.

More specifically, we will see the secret and public keys $\mathbf{S}, \mathbf{A}$ as matrices with two anti-circulant blocks of size $n \times n$, while the polynomials $\mathbf{u}, \mathbf{c}$ will be seen as column vectors of dimension n over $\mathbb{Z}_{2q}$, and vectors of polynomials $\mathbf{v}, \mathbf{y}, \mathbf{z}$ will be seen as column vectors over $\mathbb{Z}_{2q}$ of dimension $2n$.

Prover		**Verifier**
Secret Key $\mathbf{S} \in \mathcal{R}^{2\times 1}$, short ($\mathbf{S} \in \mathbb{Z}^{2n\times n}$)	$\mathbf{A}\cdot\mathbf{S} = q \mod 2q$	$\mathbf{A} = [\mathbf{a}, q-2] \in \mathcal{R}_{2q}^{1\times 2}$ ($\mathbf{A} \in \mathbb{Z}_{2q}^{n\times 2n}$)
	Commit	
Sample $\mathbf{y} \leftarrow D_\sigma^{2n}$		
Compute $\mathbf{u} = \mathbf{A}\cdot\mathbf{y} \bmod q$	$\xrightarrow{\mathbf{u}}$	
	Challenge	
	$\xleftarrow{\mathbf{c}}$	Sample $\mathbf{c} \in \mathbb{B}_\kappa^n$
Compute $\mathbf{z} = \mathbf{y} \pm \mathbf{Sc}$	**Response**	
Abort except with proba $1 - D_\sigma^{2n}(\mathbf{z})/M \cdot D_\sigma^{2n}(\mathbf{z} \pm \mathbf{Sc})$	$\xrightarrow{\mathbf{z}}$	Accept if $\|\mathbf{z}\|$ is small and $\mathbf{Az} = \mathbf{u} + q\mathbf{c}$, Accept

Figure 1: Bliss as a Σ-protocol

2.2 The original Bliss

The full description of Bliss is given in Appendix A, for our purpose we require only a simplified description of the scheme. It is designed using the so-called Fiat-Shamir paradigm [FS86] with an extra abort step as introduced in [Lyu09] to prevent leaks of secret information. The scheme can be described as a Σ-protocol (a 3-round identification protocol). It is then transformed into a signature scheme by replacing the challenge generation step by a call to a hash function (modelled as a Random Oracle) to generate the challenge $\mathbf{c}$. The Σ-protocol associated to Bliss is described in Figure 1. In this description, D_σ^{2n} denotes the discrete gaussian distribution of covariance $\sigma^2 \cdot \mathbf{Id}$ in dimension $2n$.

The repetition rate is set to $M = e^{1/(2\alpha^2)}$ and the correctness of the rejection sampling (which is required by the security proof) is ensured if for all challenges $\mathbf{c} \in \mathbb{B}_\kappa^n$, $\sigma \geq \alpha \cdot \|\mathbf{Sc}\|$. This article focuses on the improvement of the bound on $\|\mathbf{Sc}\|$, hoping to increase the parameter α and therefore accelerating the whole scheme with minimal modifications. Modifying other parameters $(\sigma, \delta, \dots)$ one could trade this speed improvement for compactness or security, but that would be a more substantial alteration of Bliss. Modifying only α and M does preserve the whole security claims of [DDLL13], as well as the optimized CDT tables precomputed in [PDG14].

The original bound. Let us recall how such bound is established in [DDLL13]. Let $\mathbf{G}$ denotes the Gram matrix associated to the secret key $\mathbf{S} \in \mathcal{S}$: $\mathbf{G} = \mathbf{S}^t \cdot \mathbf{S}$. The bound used in Bliss follows from rewriting

$$\|\mathbf{Sc}\|^2 = \mathbf{c}^t\mathbf{Gc} = \sum_{i\in\mathcal{I}_\mathbf{c}} \sum_{j\in\mathcal{I}_\mathbf{c}} \mathbf{G}_{i,j}$$

(we recall that $\mathcal{I}_\mathbf{c}$ is the indicator set of $\mathbf{c}$, defined by $\mathcal{I}_\mathbf{c} = \{i | \mathbf{c}_i = 1\}$). Such sum can be bounded independently of $\mathbf{c}$:

$$\|\mathbf{Sc}\|^2 \leq N_\kappa(\mathbf{S}) \text{ where } N_\kappa(\mathbf{S}) = \max_{\substack{I\subset\{1,\dots,n\}\\ \#I=\kappa}} \sum_{i\in I} \left(\max_{\substack{J\subset\{1,\dots,n\}\\ \#J=\kappa}} \sum_{j\in J} \mathbf{G}_{i,j} \right)$$

Name of the scheme	Bliss-0	Bliss-I	Bliss-II	Bliss-III	Bliss-IV
Security	Toy	128 bits	128 bits	160 bits	192 bits
Optimized for	Fun	Speed	Size	Security	Security
Signature size	3.3kb	5.6kb	5kb	6kb	6.5kb
dimension n	256	512	512	512	512
α	.5	1	.5	.7	.55
κ	12	23	23	30	39
Secret key N_κ-Threshold C	1.5	1.62	1.62	1.75	1.88
Repetition rate M	**7.39**	**1.65**	**7.39**	**2.77**	**5.22**

Table 1: Selected parameters of Bliss

Out of the set $\mathcal{S}_0$ of all sampleable secret keys (see Alg. 3 in Appendix A), some will be rejected, restricting the set of keys to $\mathcal{S} = \{\mathbf{S} \in \mathcal{S}_0 | N_\kappa(\mathbf{S}) \leq B\}$. This ensures the bound $\|\mathbf{Sc}\|^2 \leq B$ independently of both the choices $\mathbf{S} \in \mathcal{S}$ and $\mathbf{c} \in \mathbb{B}^n_\kappa = \mathbb{B}^n_\kappa$.

Original parameter. The (relevant) parameters of Bliss are given in Table 1 (a full table is given in Appendix A). The N_κ-threshold C defines the ratio between the bound on $\|\mathbf{Sc}\|$ and its expected value $\|\mathbf{S}\| \cdot \|\mathbf{c}\| \approx \sqrt{5 \cdot \lceil \delta n \rceil \cdot \kappa}$ (δ denotes the density of ± 1 among the ternary entries $\{0, \pm 1\}$ of the secret key $\mathbf{S}$). That is, the set of acceptable secret keys in Bliss is defined as

$$\mathcal{S} = \{\mathbf{S} \in \mathcal{S}_0 | N_\kappa(\mathbf{S}) \leq C^2 \cdot 5\lceil \delta n \rceil \cdot \kappa\}$$

for some set $\mathcal{S}_0$ implicitly defined in the Key Generation algorithm (Alg. 3).

2.3 Asymptotics of various geometric bounds

For the sake of this discussion we may drop the parameter δ and consider that the coefficients of $\mathbf{S}$ are uniformly randomly chosen from $\{\pm 1\}$. We here detail the argument behind the asymptotic claims made during the introduction.

Singular norm, non-Ring setting. The spectral theory of random matrices [Ver10] ensures that $s_1(\mathbf{S}) = \Theta(\sqrt{n})$ with overwhelming probability. This would lead to a constant $C = \Theta(1)$.

Singular norm, ring setting. Let us drop, for the sake of this discussion the binary structure of the secret and the challenge. Consider two polynomials $s, c \in \mathcal{R} = \mathbb{R}[X]/\Phi_{2n}(X)$ where s is drawn with spherical gaussian distribution in the coefficient representation ($s = \sum f_j X_j, f_j \leftarrow \chi$). Now consider the complex representation (a.k.a. the Fourier transform of s) $\hat{s}$ of s ($\hat{s}_j = s(e^{\pi(2j+1)\imath/n})$), we remind that the Fourier transform is a scaled orthogonal map: $\|\hat{x}\| = \sqrt{n} \cdot \|x\|$. Because in the complex embedding multiplication occurs component-wise, that is $\widehat{sc} = \hat{s} \odot \hat{c}$, we conclude that $s_1(s) = \|\hat{s}\|_\infty$. We expect from order statistic theory $\|\hat{s}\|_\infty = O(\sqrt{n \log n})$ as the maximum of n many Gaussian of variance $\sqrt{n}$ (see [Roy82]). This would lead to $C = O(\sqrt{\log n})$.

The N_κ-norm, Ring Setting. A small modification of the proof of [DDLL13, Prop. 4.1 of the full version] can establish that

$$N_\kappa(\mathbf{S}) \leq (5\lceil \delta n \rceil + 1)\kappa + \kappa^2 \sqrt{n\delta} \cdot \omega(\sqrt{\log n})$$

with overwhelming probability. Note that the only constraint on κ is that $\mathbf{c} \leftarrow \mathbb{B}_\kappa^n$ has at least $\Theta(n)$ bits of entropy, requiring $\kappa \geq \Theta(n/\log n)$. If the second term were negligible one would be able to choose $C = 1 + o(1)$. While it is not asymptotically negligible, this second term remains reasonably small for the parameter of BLISS explaining why the constant C could be chosen as small as 1.5.

3 The Sign Choices algorithm and its Geometry

As detailed in the introduction, we will exploit ternary representation in $\mathbb{Z}^n$ of challenge vectors modulo 2, by individually negating some coordinates of the binary vector $\mathbf{c}$ in order to greedily minimize the norm of $\|\mathbf{Sc}\|$. More specifically, we rely on the identity:

$$\|\mathbf{a} + \mathbf{b}\|^2 = \|\mathbf{a}\|^2 + \|\mathbf{b}\|^2 + 2\,\langle \mathbf{a}\,,\,\mathbf{b}\rangle \tag{1}$$

which implies that either $\|\mathbf{a} + \mathbf{b}\|^2$ or $\|\mathbf{a} - \mathbf{b}\|^2$ is less than $\|\mathbf{a}\|^2 + \|\mathbf{b}\|^2$. Precisely,

$$\|\mathbf{a} - \varsigma\mathbf{b}\|^2 \leq \|\mathbf{a}\|^2 + \|\mathbf{b}\|^2 \qquad \text{where } \varsigma = \mathrm{sgn}(\langle \mathbf{a}\,,\,\mathbf{b}\rangle) \in \{\pm 1\} \tag{2}$$

where the sign function takes (arbitrarily) the value 1 on input 0.

Algorithm 1: $\mathsf{GreedySC}(\mathbf{S}, \mathbf{c})$, The Greedy Sign Choices Algorithm.

Input: a matrix $\mathbf{S} = [\mathbf{s}_1 \dots \mathbf{s}_n] \in \mathbb{Z}^{m\times n}$ and a binary vector $\mathbf{c} \in \mathbb{B}_\kappa^n$
Output: $\mathbf{v} = \mathbf{Sc}'$ for some $\mathbf{c}' \equiv \mathbf{c} \bmod 2$ such that $\|\mathbf{Sc}'\|^2 \leq \|\mathbf{S}\|^2 \cdot \|\mathbf{c}\|^2 = \|\mathbf{S}\|^2 \cdot \kappa$.

```
v ← 0 ∈ Z^m
for i ∈ I_c do
    ς_i ← sgn(⟨v , s_i⟩)
    v ← v − ς_i · s_i
end for
return v
```

Theorem 1 (Correctness of the Greedy Sign Choices Algorithm)**.** *On inputs* $\mathbf{S} \in \mathbb{Z}^{m\times n}$ *and* $\mathbf{c} \in \mathbb{B}_\kappa^n$*, the algorithm outputs* $\mathbf{v} = \mathsf{GreedySC}(\mathbf{S}, \mathbf{c})$ *such that* $\mathbf{v} = \mathbf{Sc}'$ *for some ternary vector* $\mathbf{c}' \equiv \mathbf{c} \bmod 2$ *and*

$$\|\mathbf{Sc}'\|^2 \leq \|\mathbf{S}\|^2 \cdot \|\mathbf{c}\|^2 = \|\mathbf{S}\|^2 \cdot \kappa.$$

Proof. We prove an invariant of the for-loop of $\mathsf{GreedySC}$, assuming the set $\mathcal{I}_\mathbf{c}$ is visited in some fixed order. We consider the subset $\mathcal{J}_k \subset \mathcal{I}_\mathbf{c}$ of indexes i that have been visited after k steps and set

$$\mathbf{c}'_k = -\sum_{i\in\mathcal{J}_k} \varsigma_i \cdot \mathbf{e}_i \qquad (\mathbf{e}_i \in \mathbb{Z}^n \text{ being the i-th canonical unit vector}).$$

We will prove the following invariant: $\mathbf{c}'_k$ is a ternary vector such that $\mathcal{I}_{\mathbf{c}'_k} = \mathcal{J}_k$ and $\mathbf{v}_k = \mathbf{S}\cdot\mathbf{c}'_k$ verifies $\|\mathbf{v}_k\|^2 \leq \|\mathbf{S}\| \cdot k$. It is trivially verified for $k = 0$. For the induction, first decompose $\mathcal{J}_{k+1} = \mathcal{J}_k \cup \{i\}$ where $i \in \mathcal{I}_\mathbf{c} \setminus \mathcal{J}_k$. Following inequality (2) we derive:

$$\|\mathbf{v}_{k+1}\|^2 \leq \|\mathbf{v}_k\|^2 + \|\mathbf{s}_i\|^2 \leq \|\mathbf{S}\|^2 \cdot k + \|\mathbf{S}\|^2 \leq \|\mathbf{S}\|^2 \cdot (k+1).$$

We conclude that the output $\mathbf{v} = \mathbf{v}_\kappa = \mathbf{Sc}'_\kappa$ verifies $\|\mathbf{v}\|^2 \leq \|\mathbf{S}\|^2 \cdot \kappa$ and that $\mathbf{c}' = \mathbf{c}'_\kappa$ is a ternary vector verifying $\mathcal{I}_{\mathbf{c}'} = \mathcal{I}_\mathbf{c}$. □

Efficiency. One may note the algorithm is not technically running in quasilinear time: it requires $O(m\kappa)$ integer additions, where $m = 2n$ and $\kappa = \Omega(n/\log n)$ to ensure that $\mathbf{c} \leftarrow \mathbb{B}_\kappa^n$ contains at least $\Theta(n)$ bits of entropy. Yet in practice the parameter κ is quite small (at most $\kappa = 39$ against $n = 512$ for Bliss-IV) making the sparse subset-sum algorithm to compute $\mathbf{Sc}$ faster than the Number Theoretic Transform based algorithm, especially considering that the entries of $\mathbf{S}$ are quite small. The implementation [DL] of Bliss indeed use a 64-bits vectorized subset-sum algorithm with 8-bits chunks.

Still, this new algorithm will be substantially slower: we need twice as many operations because of the added inner product, and those require computing with larger numbers. We need to be careful to avoid making this part of the algorithm way too costly in practice. The following vectorization should be enough to keep the cost of GreedySC much smaller than the Gaussian sampling of $\mathbf{y}_1, \mathbf{y}_2$ and the NTT product $\mathbf{a}_1 \cdot \mathbf{y}_1$.

Vectorized Implementation with 16-bits chunks. For the parameters of Bliss, we always have $\|\mathbf{S}\|_\infty \leq 5$ (see Algorithm 4 in Appendix A) which guarantees that $\|\mathbf{v}\|_\infty \leq 5\kappa \leq 195$ during the whole algorithm: the entries of $\mathbf{v}$ can definitely be stored as 16-bit signed integers. Additionally, any partial sum during the computation of $\langle \mathbf{v}\,, \mathbf{s}_i \rangle$ must be less in absolute value than $m = \|\mathbf{v}\| \cdot \|\mathbf{S}\| \leq \|\mathbf{S}\|^2 \cdot \sqrt{\kappa}$. For all parameter sets Bliss-0 to Bliss-IV one easily checks that m is much less than 2^{15}. The whole GreedySC can therefore benefit from vectorized instructions with 16-bits signed chunks.

4 The modified scheme and instantiations

4.1 Modifications of the scheme

We define a modified version Bliss-b of the scheme Bliss from [DDLL13]. The modifications to the original scheme are listed below:

- We remove the constant C and the ad-hoc norm N_κ from the definitions
- A constant
$$P_{\max} = \begin{cases} (5\lceil \delta_1 n \rceil + 5) \cdot \kappa & \text{if } \delta_2 = 0 \\ (5\lceil \delta_1 n \rceil + 20\lceil \delta_2 n \rceil + 9) \cdot \kappa & \text{otherwise} \end{cases}$$
is defined (syntactic sugar). The output product $\mathbf{Sc}' = \mathsf{GreedySC}(\mathbf{S}, \mathbf{c})$ verifies $\|\mathbf{Sc}'\|^2 \leq P_{\max}$ since $\|\mathbf{S}\|^2 = 5(\lceil \delta_1 n \rceil + 4\lceil \delta_2 n \rceil) + 4\mathbf{g}_0 + 1$.
- Line 3 of Algorithm 3 is removed: keys are no longer rejected during Key Generation.
- The Signature algorithm (Algorithm 4) is replaced by Algorithm 2, described below.

One may note that leaking the sign choices made in $\mathbf{c}'$ during the algorithm GreedySC would reveal information on the secret key $\mathbf{S}$. Fortunately, such information is only carried by $\mathbf{v}$, and the rejection step is exactly designed to perfectly hide $\mathbf{v}$ assuming $\|\mathbf{v}\| \leq P_{\max}$.

Algorithm 2: Modified Bliss Signature Algorithm

Input: Message μ, public key $\mathbf{A} = (\mathbf{a}_1, q-2) \in \mathcal{R}_{2q}^{1\times 2}$, secret key $\mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2)^t \in \mathcal{R}_{2q}^{2\times 1}$

Output: A signature $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$ of the message μ

1: $\mathbf{y}_1, \mathbf{y}_2 \leftarrow D_{\mathbb{Z}^n,\sigma}$
2: $\mathbf{u} = \zeta \cdot \mathbf{a}_1 \cdot \mathbf{y}_1 + \mathbf{y}_2 \bmod 2q$
3: $\mathbf{c} \leftarrow H(\lfloor \mathbf{u} \rceil_d \bmod p, \mu)$
4: $(\mathbf{v}_1, \mathbf{v}_2) \leftarrow \mathsf{GreedySC}(\mathbf{S}, \mathbf{c})$
5: Choose a random bit b
6: $(\mathbf{z}_1, \mathbf{z}_2) \leftarrow (\mathbf{y}_1, \mathbf{y}_2) + (-1)^b \cdot (\mathbf{v}_1, \mathbf{v}_2)$
7: **Continue** with probability $1 \Big/ \left(M \exp\left(-\frac{\|\mathbf{v}\|^2}{2\sigma^2}\right) \cosh\left(\frac{\langle \mathbf{z}, \mathbf{v}\rangle}{\sigma^2}\right)\right)$ otherwise **restart**
8: $\mathbf{z}_2^\dagger \leftarrow (\lfloor \mathbf{u} \rceil_d - \lfloor \mathbf{u} - \mathbf{z}_2 \rceil_d) \bmod p$
9: **Output** $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$

4.2 New parameters and benchmarks

We left all parameters related to security unmodified, so that the security claims of [DDLL13] are preserved, our modifications only affect the efficiency of the scheme. we recall that the rejection rate is defined as $M = \exp(1/(2\alpha^2))$. The new value of α is given by $\alpha = \sigma/P_{\max}$. We obtain a theoretical speed-up ranging from 1.4 to 3.4 as detailed in Table 2. The Key Generation is also accelerated by a factor 5 to 10 since secret keys are not rejected according to N_κ anymore. Our implementation conforms to those predictions considering a small slowdown related to the added cost of the greedy algorithm: the actual speed-up factor varies from 1.2 (Bliss-bI) to 2.8 (Bliss-bII). Table 3 compares the running time of Bliss and Bliss-b.

Scheme	Bliss-b0	Bliss-bI	Bliss-bII	Bliss-bIII	Bliss-bIV
Security	Toy	128 bits	128 bits	160 bits	192 bits
Optimized for	Fun	Speed	Size	Security	Security
Signature size	3.3kb	5.6kb	5kb	6kb	6.5kb
dimension n	256	512	512	512	512
α	.748	1.610	.801	1.216	1.027
Repetition rate M	**2.44**	**1.21**	**2.18**	**1.40**	**1.61**
Predicted Speed-up	**×3.0**	**×1.4**	**×3.4**	**×2.0**	**×3.3**

Table 2: Parameters of the modified scheme Bliss-b

Scheme	Bliss-0	Bliss-I	Bliss-II	Bliss-III	Bliss-IV
Sign Run Time	256 μs	154 μs	520 μs	240 μs	419 μs
Scheme	Bliss-b0	Bliss-bI	Bliss-bII	Bliss-bIII	Bliss-bIV
Sign Run Time	108 μs	128 μs	185 μs	146 μs	167 μs
Speed-up	**×2.4**	**×1.2**	**×2.8**	**×1.6**	**×2.5**

Table 3: Running Time in microseconds averaged over 10 000 signatures. (Intel Core @ 3.40GHz).

Backward and Forward compatibility. Signatures are both backward and forward compatible, that is signature generated using Bliss-b will also be valid for Bliss and reciprocally, since both schemes are corrects and have the same verification algorithm. Secret keys are only forward compatible, that is secret keys generated by Bliss can be used in Bliss-b and do benefit from the speed-up as well. Yet secret keys generated by Bliss-b should not be used in Bliss: the condition $\sigma \geq \alpha \cdot \|\mathbf{Sc}\|$ may not always hold leading to information leakage.

Acknowledgements

The authors wish to thank V. Lyubashevsky, T. Lepoint, and J.C. Deneuville for helpful comments. This research was supported in part by the DARPA PROCEED program and NSF grant CNS-1117936. Opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA or NSF.

References

[AD97] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *29th ACM STOC*, pages 284–293. ACM Press, May 1997.

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *28th ACM STOC*, pages 99–108. ACM Press, May 1996.

[BG14] Shi Bai and Steven D. Galbraith. An improved compression technique for signatures based on learning with errors. In Josh Benaloh, editor, *CT-RSA 2014*, volume 8366 of *LNCS*, pages 28–47. Springer, February 2014.

[Boy10] Xavier Boyen. Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more. In Phong Q. Nguyen and David Pointcheval, editors, *PKC 2010*, volume 6056 of *LNCS*, pages 499–517. Springer, May 2010.

[CHKP12] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. *Journal of Cryptology*, 25(4):601–639, October 2012.

[DDLL13] Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. Lattice signatures and bimodal gaussians. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 40–56. Springer, August 2013.

[DL] Léo Ducas and Tancrède Lepoint. *A Proof-of-concept Implementation of BLISS*. Available under the CeCILL License at `http://bliss.di.ens.fr`.

[DM14] Léo Ducas and Daniele Micciancio. Improved short lattice signatures in the standard model. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 335–352. Springer, August 2014.

[DN12] Léo Ducas and Phong Q. Nguyen. Learning a zonotope and more: Cryptanalysis of NTRUSign countermeasures. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 433–450. Springer, December 2012.

[DPL14] Leo Ducas, Thomas Prest, and Vadim Lyubashevsky. Efficient identity-based encryption over ntru lattices. 2014. To be Published at Asiacrypt 2014.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, August 1986.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 169–178. ACM Press, May / June 2009.

[GLP12] Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In Emmanuel Prouff and Patrick Schaumont, editors, *CHES 2012*, volume 7428 of *LNCS*, pages 530–547. Springer, September 2012.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008.

[GVW13] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 545–554. ACM Press, June 2013.

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NTRU: A ring-based public key cryptosystem. In *Algorithmic Number Theory – Proc. ANTS-III*, volume 1423 of *Lecture Notes in Computer Science*, pages 267–288. Springer, 1998.

[LMPR08] Vadim Lyubashevsky, Daniele Micciancio, Chris Peikert, and Alon Rosen. SWIFFT: A modest proposal for FFT hashing. In Kaisa Nyberg, editor, *FSE 2008*, volume 5086 of *LNCS*, pages 54–72. Springer, February 2008.

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23. Springer, May 2010.

[Lyu09] Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616. Springer, December 2009.

[Lyu12] Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 738–755. Springer, April 2012.

[MBDG14] Carlos Aguilar Melchor, Xavier Boyen, Jean-Christophe Deneuville, and Philippe Gaborit. Sealing the leak on classical ntru signatures. In *Post-Quantum Cryptography*, pages 1–21. Springer, 2014.

[Mic02] Daniele Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions from worst-case complexity assumptions. In *43rd FOCS*, pages 356–365. IEEE Computer Society Press, November 2002.

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, April 2012.

[NR06] Phong Q. Nguyen and Oded Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 271–288. Springer, May / June 2006.

[PDG14] Thomas Pöppelmann, Léo Ducas, and Tim Güneysu. Enhanced lattice-based signatures on reconfigurable hardware. In Lejla Batina and Matthew Robshaw, editors, *CHES 2014*, volume 8731 of *LNCS*, pages 353–370. Springer, September 2014.

[Roy82] J. P. Royston. Algorithm AS 177: Expected Normal Order Statistics (Exact and Approximate). *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 31(2):161–165, 1982.

[SS11] Damien Stehlé and Ron Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 27–47. Springer, May 2011.

[Ver10] Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *CoRR*, abs/1011.3027, 2010. `http://www-personal.umich.edu/~romanv/papers/non-asymptotic-rmt-plain.pdf`.

A Full Description of Original Bliss and parameters

Algorithm 3: Bliss Key Generation

Output: Key pair $(\mathbf{A}, \mathbf{S})$ such that $\mathbf{AS} = q \bmod 2q$

1: Choose $\mathbf{f}, \mathbf{g}$ as uniform polynomials with exactly $d_1 = \lceil \delta_1 n \rceil$ entries in $\{\pm 1\}$ and $d_2 = \lceil \delta_2 n \rceil$ entries in $\{\pm 2\}$
2: $\mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2)^t \leftarrow (\mathbf{f}, 2\mathbf{g}+1)^t$ *// Implicitly defining the set* $\mathcal{S}_0$
3: **if** $N_\kappa(\mathbf{S}) \geq C^2 \cdot 5 \cdot (\lceil \delta_1 n \rceil + 4\lceil \delta_2 n \rceil) \cdot \kappa$ **then** restart
4: $\mathbf{a}_q = (2\mathbf{g}+1)/\mathbf{f} \bmod q$ (**restart** if $\mathbf{f}$ is not invertible)
5: **Output**$(\mathbf{A}, \mathbf{S})$ where $\mathbf{A} = (2\mathbf{a}_q, q-2) \bmod 2q$

Algorithm 4: Bliss Signature Algorithm

Input: Message μ, public key $\mathbf{A} = (\mathbf{a}_1, q-2) \in \mathcal{R}_{2q}^{1\times 2}$, secret key $\mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2)^t \in \mathcal{R}_{2q}^{2\times 1}$

Output: A signature $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$ of the message μ

1: $\mathbf{y}_1, \mathbf{y}_2 \leftarrow D_{\mathbb{Z}^n, \sigma}$
2: $\mathbf{u} = \zeta \cdot \mathbf{a}_1 \cdot \mathbf{y}_1 + \mathbf{y}_2 \bmod 2q$
3: $\mathbf{c} \leftarrow H(\lfloor \mathbf{u} \rceil_d \bmod p, \mu)$
4: Choose a random bit b
5: $\mathbf{z}_1 \leftarrow \mathbf{y}_1 + (-1)^b \mathbf{s}_1 \mathbf{c}$; $\quad \mathbf{z}_2 \leftarrow \mathbf{y}_2 + (-1)^b \mathbf{s}_2 \mathbf{c}$
6: **Continue** with probability $1 \Big/ \left(M \exp\left(-\frac{\|\mathbf{Sc}\|^2}{2\sigma^2} \right) \cosh\left(\frac{\langle \mathbf{z}, \mathbf{Sc} \rangle}{\sigma^2} \right) \right)$ otherwise **restart**
7: $\mathbf{z}_2^\dagger \leftarrow (\lfloor \mathbf{u} \rceil_d - \lfloor \mathbf{u} - \mathbf{z}_2 \rceil_d) \bmod p$
8: **Output** $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$

Algorithm 5: Bliss Verification Algorithm

Input: Message μ, public key $\mathbf{A} = (\mathbf{a}_1, q-2) \in \mathcal{R}_{2q}^{1\times 2}$, signature $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$

Output: Accept or Reject the signature

1: **if** $\|(\mathbf{z}_1 | 2^d \cdot \mathbf{z}_2^\dagger)\|_2 > B_2$ **then** Reject
2: **if** $\|(\mathbf{z}_1 | 2^d \cdot \mathbf{z}_2^\dagger)\|_\infty > B_\infty$ **then** Reject
3: Accept iff $\mathbf{c} = H\big(\lfloor \zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot q \cdot \mathbf{c} \rceil_d + \mathbf{z}_2^\dagger \bmod p, \mu\big)$

Name of the scheme	**BLISS-0**	**BLISS-I**	**BLISS-II**	**BLISS-III**	**BLISS-IV**
Security	Toy ($\leq$ 60 bits)	128 bits	128 bits	160 bits	192 bits
Optimized for	Fun	Speed	Size	Security	Security
n	256	512	512	512	512
Modulus q	7681	12289	12289	12289	12289
Secret key densities δ_1, δ_2	.55 , .15	.3 , 0	.3 , 0	.42 , .03	.45, .06
Gaussian standard deviation σ	100	215	107	250	271
α	.5	1	.5	.7	.55
κ	12	23	23	30	39
Secret key N_κ-Threshold C	1.5	1.62	1.62	1.75	1.88
Dropped bits d in $\mathbf{z}_2$	5	10	10	9	8
Verification thresholds B_2, B_∞	2492, 530	12872, 2100	11074, 1563	10206,1760	9901, 1613
Repetition rate	7.4	1.6	7.4	2.8	5.2
Entropy of challenge $\mathbf{c} \in \mathbb{B}_\kappa^n$	66 bits	132 bits	132 bits	161 bits	195 bits
Signature size	**3.3kb**	**5.6kb**	**5kb**	**6kb**	**6.5kb**
Secret key size	1.5kb	2kb	2kb	3kb	3kb
Public key size	3.3kb	7kb	7kb	7kb	7kb