Introduction to HAETAE:

Post-quantum Signature Scheme based on Hyperball Bimodal Rejection Sampling

Jung Hee Cheon[1,2], **Hyeongmin Choe**[1], Julien Devevey[3], Tim Güneysu[4], Dongyeon Hong[2], Markus Krausz[4], Georg Land[4], Junbum Shin[2], Damien Stehlé[3,5], MinJune Yi[1,2]

[1]Seoul National University, [2]CryptoLab Inc., [3]École Normale Supérieure de Lyon, [4]Ruhr Universität Bochum, [5]Institut Universitaire de France

February 22, 2023

Table of Contents

HAETAE

- Digital signature scheme
- Secure against quantum attacks
  - based on **lattice hard problems** MLWE and MSIS
  - follows **Fiat-Shamir with aborts** framework, secure in QROM
- Simple but short
  - simpler than Falcon[1] & shorter than Dilithium[1]
  - optimal rejection rate with simple rejection condition
- Design rationale
  - **Fiat-Shamir with aborts** framework
  - using **Bimodal** rejection sampling
  - randomness sampling from **Hyperball** distribution

[1] NIST 2022 PQC signature standards

HAETAE

- Digital signature scheme
- Secure against quantum attacks
  - based on **lattice hard problems** MLWE and MSIS
  - follows **Fiat-Shamir with aborts** framework, secure in QROM
- Simple but short
  - simpler than Falcon[1] & shorter than Dilithium[1]
  - optimal rejection rate with simple rejection condition
- Design rationale
  - **Fiat-Shamir with aborts** framework
  - using **Bimodal** rejection sampling
  - randomness sampling from **Hyperball** distribution

[1] NIST 2022 PQC signature standards

HAETAE

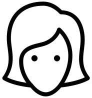
- Digital signature scheme
- Secure against quantum attacks
  - based on **lattice hard problems** MLWE and MSIS
  - follows **Fiat-Shamir with aborts** framework, secure in QROM
- Simple but short
  - simpler than Falcon[1] & shorter than Dilithium[1]
  - optimal rejection rate with simple rejection condition
- Design rationale
  - **Fiat-Shamir with aborts** framework
  - using **Bimodal** rejection sampling
  - randomness sampling from **Hyperball** distribution

[1] NIST 2022 PQC signature standards

Digital signatures

Conventional signatures work as:

Alice



wants to sign
on a document

Bob

wants to verify
that Alice signed
on the document

Digital signatures

Conventional signatures work as:

Alice

wants to sign
on a document

Bob

wants to verify
that Alice signed
on the document

Digital signatures work as:

$(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}$ and broadcast vk

Alice (knows sk)

signature $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$

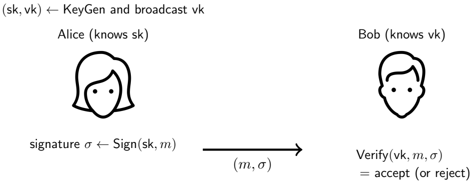
(m, σ)

Bob (knows vk)

$\mathsf{Verify}(\mathsf{vk}, m, \sigma)$
$= \mathsf{accept}$ (or reject)

Digital signatures

Digital signatures work as:



Necessary properties:

- **Correctness**:

$$\mathsf{Verify}(\mathsf{vk}, m, \mathsf{Sign}(\mathsf{sk}, m)) = \mathsf{accept}$$

- **Unforgeability**: No one else than Alice can make a new signature. More formally,

  For a given verification key and some message-signature pairs, no adversary can forge a new valid signature.

1. Brief Introduction to HAETAE

2. Preliminaries:
   - Digital signatures
   - Lattice hard problems and lattice-based signatures
   - Details of "Fiat-Shamir with aborts"
     - Rejection sampling
     - Bimodal rejection sampling

3. Dive into HAETAE:
   - HAETAE, in theory
     - Hyperball bimodal rejection sampling
     - Compression techniques
   - Implementation
     - Parameters and concrete security
     - Reference implementation

4. Upcoming updates!

Lattice hard problems

Lattice-based cryptography is the generic term for constructions of cryptographic primitives that involve lattices ... are currently important candidates for post-quantum cryptography. - *Wikipedia*

Lattice-based cryptography bases its security on lattice hard problems, which are studied for the last 20–30 years with strong theoretical backgrounds:

- SVP and GapSVP_λ are NP-hard for randomized **reductions** on some limited parameters [Ajt96, HR07].
- worst-case to average-case **reductions** [Ajt96], meaning that worst-case problems are not harder than average-case problems.
- Useful hard problems: NTRU, LWE, SIS, MLWE, MSIS, etc : hard problems for random instances.

Lattice hard problems

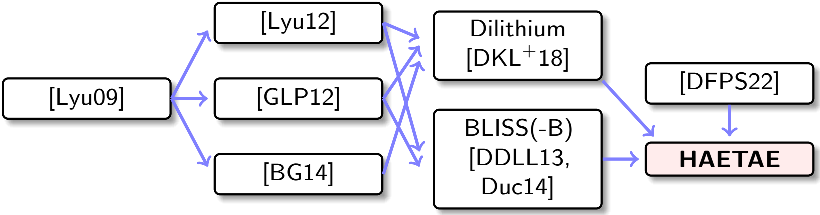
Lattice-based cryptography is the generic term for constructions of cryptographic primitives that involve lattices ... are currently important candidates for post-quantum cryptography. - *Wikipedia*

Lattice-based cryptography bases its security on lattice hard problems, which are studied for the last 20–30 years with strong theoretical backgrounds:

- SVP and GapSVP$_\lambda$ are NP-hard for randomized **reductions** on some limited parameters [Ajt96, HR07].
- worst-case to average-case **reductions** [Ajt96], meaning that worst-case problems are not harder than average-case problems.
- Useful hard problems: NTRU, LWE, SIS, MLWE, MSIS, etc : hard problems for random instances.

Lattice-based signatures

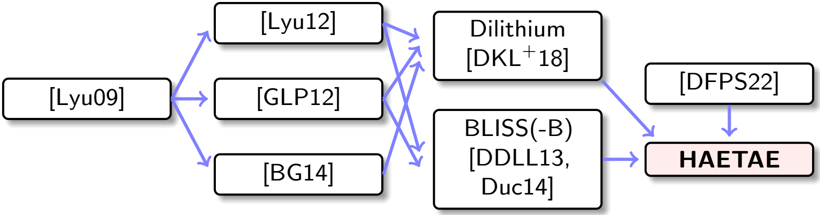
Fiat-Shamir with abort



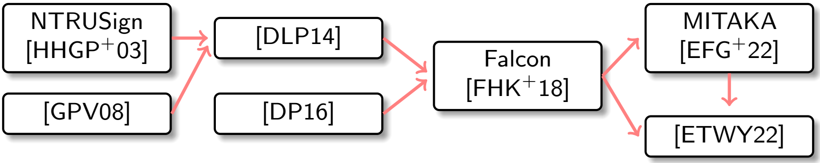
Hash-and-Sign

Lattice-based signatures

Fiat-Shamir with abort



Hash-and-Sign



Lattice-based signatures

Fiat-Shamir with abort:

KeyGen : output $(\mathsf{sk} = \mathrm{s},\ \mathsf{vk} = \mathbf{t})$, where $\mathbf{t} = \mathbf{A}\mathrm{s} \bmod q$ and s is short.

Sign$(\mathsf{sk} = \mathrm{s},\ m)$: for short $\mathbf{y}$, output $(c = H(\mathbf{Ay} \bmod q,\ m),\ \mathbf{z} = \mathbf{y} + c\mathrm{s})$ via rejection sampling.

Verify$(\mathsf{vk} = \mathbf{A},\ m,\ \mathrm{s})$: check whether $c = H(\mathbf{Az} - c\mathbf{t} \bmod q, m)$ and $\mathbf{z}$ is short.

Correctness:

- First, $\mathbf{y}$ and s are short. Since $c = H(\cdot)$ is binary, $c\mathrm{s}$ is also short. Thus, $\mathbf{z} = \mathbf{y} + c\mathrm{s}$ is short.
- It holds that $\mathbf{Az} - c\mathbf{t} = \mathbf{A}(\mathbf{y} + c\mathbf{s}) - c\mathbf{t} = \mathbf{Ay} \bmod q$ since $\mathbf{As} = \mathbf{t} \bmod q$.

Fiat-Shamir with aborts

Basic “Fiat-Shamir with aborts” framework [Lyu09, Lyu12]

KeyGen : output (sk $= \mathbf{s}$, vk $= \mathbf{t}$), where $\mathbf{t} = \mathbf{A}\mathbf{s} \bmod q$ and $\mathbf{s}$ is short.

Sign(sk $= \mathbf{s}$, m) : for short $\mathbf{y}$, output $(c = H(\mathbf{A}\mathbf{y} \bmod q,\ m),\ \mathbf{z} = \mathbf{y} + c\mathbf{s})$ via rejection sampling.

Verify(vk $= \mathbf{A}$, m, $\mathbf{s}$) : check whether $c = H(\mathbf{A}\mathbf{z} - c\mathbf{t} \bmod q, m)$ and $\mathbf{z}$ is short.

Signature schemes following the “Fiat-Shamir with aborts” framework have well-studied **quantum security** [KLS18].

Unforgeability:

- Key security: vk does not leak sk (LWE).
- Zero-knowledge (HVZK): $(c, \mathbf{z})$ does not leak sk (rejection sampling).
- (Special) Soundness: cannot convince Bob without sk (SIS).

Fiat-Shamir with aborts

Basic “Fiat-Shamir with aborts” framework [Lyu09, Lyu12]

KeyGen : output $(\mathsf{sk} = \mathbf{s},\ \mathsf{vk} = \mathbf{t})$, where $\mathbf{t} = \mathbf{A}\mathbf{s} \bmod q$ and $\mathbf{s}$ is short.

Sign$(\mathsf{sk} = \mathbf{s},\ m)$: for short $\mathbf{y}$, output $(c = H(\mathbf{A}\mathbf{y} \bmod q,\ m),\ \mathbf{z} = \mathbf{y} + c\mathbf{s})$ via rejection sampling.

Verify$(\mathsf{vk} = \mathbf{A},\ m,\ \mathbf{s})$: check whether $c = H(\mathbf{A}\mathbf{z} - c\mathbf{t} \bmod q, m)$ and $\mathbf{z}$ is short.

Signature schemes following the “Fiat-Shamir with aborts” framework have well-studied **quantum security** [KLS18].

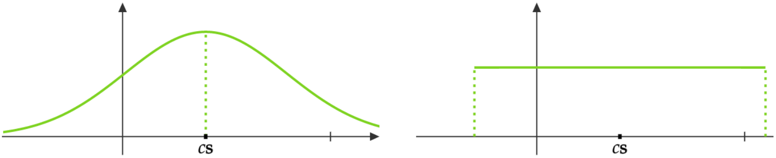
Unforgeability:

- Key security: vk does not leak sk (LWE).
- Zero-knowledge (HVZK): $(c, \mathbf{z})$ does not leak sk (rejection sampling).
- (Special) Soundness: cannot convince Bob without sk (SIS).

Rejection sampling

Naïvely, $(c, \mathbf{z} = \mathbf{y} + c\mathbf{s})$ can leak some partial information of $\mathbf{s}$.

Suppose we have an ultimate number of pairs $(c, \mathbf{z} = \mathbf{y} + c\mathbf{s})$ so that we can collect $\mathbf{z}$'s for the same c. Then the distribution of $\mathbf{z}$ can be drawn as:



depending on the distribution of $\mathbf{y}$ (e.g. discrete Gaussian or uniform).

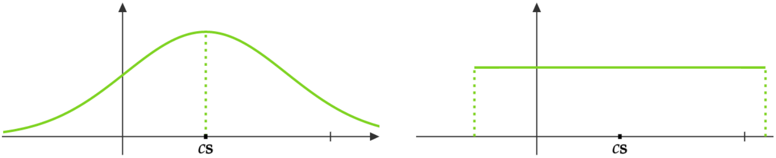
The distribution leaks $c\mathbf{s}$, i.e. the secret key sk.

$\Longrightarrow$ Rejection sampling prevents this leakage.

Rejection sampling

Naïvely, $(c, \mathbf{z} = \mathbf{y} + c\mathbf{s})$ can leak some partial information of $\mathbf{s}$.

Suppose we have an ultimate number of pairs $(c, \mathbf{z} = \mathbf{y} + c\mathbf{s})$ so that we can collect $\mathbf{z}$'s for the same c. Then the distribution of $\mathbf{z}$ can be drawn as:



depending on the distribution of $\mathbf{y}$ (e.g. discrete Gaussian or uniform).

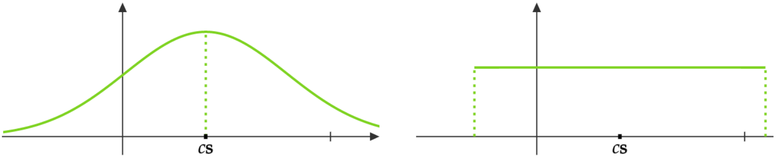
The distribution leaks $c\mathbf{s}$, i.e. the secret key sk.

$\Longrightarrow$ Rejection sampling prevents this leakage.

Rejection sampling

Naïvely, $(c, \mathbf{z} = \mathbf{y} + c\mathbf{s})$ can leak some partial information of $\mathbf{s}$.

Suppose we have an ultimate number of pairs $(c, \mathbf{z} = \mathbf{y} + c\mathbf{s})$ so that we can collect $\mathbf{z}$'s for the same c. Then the distribution of $\mathbf{z}$ can be drawn as:



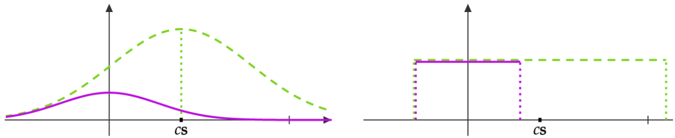
depending on the distribution of $\mathbf{y}$ (e.g. discrete Gaussian or uniform).

The distribution leaks $c\mathbf{s}$, i.e. the secret key sk.

$\Longrightarrow$ Rejection sampling prevents this leakage.

Rejection sampling

Rejection sampling rejects the pair $(c, \mathbf{z})$ with a certain probability[2], then restarts. This makes the distribution of signature independent to sk:



If $p_t(\mathbf{x}) \leq M \cdot p_s(\mathbf{x})$ for almost all $x = (c, \mathbf{z})$, the followings are identical:

i) sampling from source distribution p_s with rejection sampling ($\mathcal{A}^{\text{real}}$)

ii) sampling from target distribution p_t and reject with probability $\frac{1}{M}$ ($\mathcal{A}^{\text{ideal}}$)

$\mathcal{A}^{\text{real}}$:

1: $\mathbf{x} \leftarrow p_s$

2: Return $\mathbf{x}$ with probability $\min\left(\frac{p_t(\mathbf{x})}{M \cdot p_s(\mathbf{x})}, 1\right)$

3: Else repeat 1–2

$\mathcal{A}^{\text{ideal}}$:

1: $\mathbf{x} \leftarrow p_t$

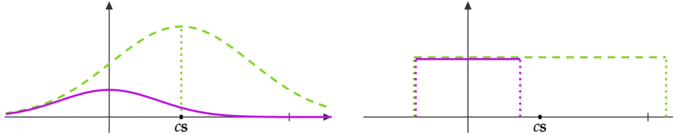
2: Return $\mathbf{x}$ with probability $\frac{1}{M}$

3: Else repeat 1–2

[2] a function of $(c, \mathbf{z})$.

Rejection sampling

Rejection sampling rejects the pair $(c, \mathbf{z})$ with a certain probability[2], then restarts. This makes the distribution of signature independent to sk:



If $p_{\mathsf{t}}(\mathbf{x}) \leq M \cdot p_{\mathsf{s}}(\mathbf{x})$ for almost all $x = (c, \mathbf{z})$, the followings are identical:

i) sampling from source distribution p_{s} with rejection sampling ($\mathcal{A}^{\mathsf{real}}$)

ii) sampling from target distribution p_{t} and reject with probability $\frac{1}{M}$ ($\mathcal{A}^{\mathsf{ideal}}$)

$\mathcal{A}^{\mathsf{real}}$:	$\mathcal{A}^{\mathsf{ideal}}$:
1: $\mathbf{x} \leftarrow p_{\mathsf{s}}$	1: $\mathbf{x} \leftarrow p_{\mathsf{t}}$
2: Return $\mathbf{x}$ with probability $\min\left(\frac{p_{\mathsf{t}}(\mathbf{x})}{M \cdot p_{\mathsf{s}}(\mathbf{x})}, 1\right)$	2: Return $\mathbf{x}$ with probability $\frac{1}{M}$
3: Else repeat 1–2	3: Else repeat 1–2

[2] a function of $(c, \mathbf{z})$.

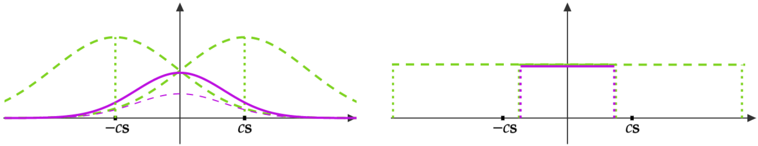
Bimodal rejection sampling

The run-time of rejection sampling depends on the constant M ($\approx$ ratio between green and purple areas).

To decrease M, [DDLL13] modified $\mathbf{z} = \mathbf{y} + c\mathbf{s}$ to

$$\mathbf{z} = \mathbf{y} + (-1)^b c\mathbf{s}$$

with modulus $2q$ instead of q:



Note that M does not change if $\mathbf{y}$ is chosen from the uniform interval.

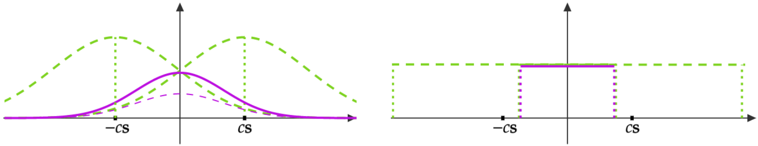
Bimodal rejection sampling

The run-time of rejection sampling depends on the constant M ($\approx$ ratio between green and purple areas).

To decrease M, [DDLL13] modified $\mathbf{z} = \mathbf{y} + c\mathbf{s}$ to

$$\mathbf{z} = \mathbf{y} + (-1)^b c\mathbf{s}$$

with modulus $2q$ instead of q:



Note that M does not change if $\mathbf{y}$ is chosen from the uniform interval.

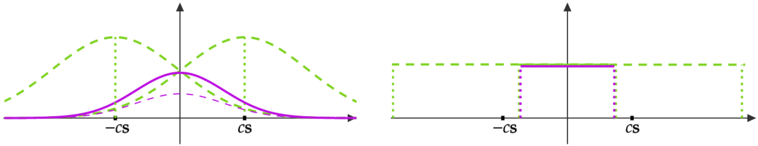
Bimodal rejection sampling

The run-time of rejection sampling depends on the constant M ($\approx$ ratio between green and purple areas).

To decrease M, [DDLL13] modified $\mathbf{z} = \mathbf{y} + c\mathbf{s}$ to

$$\mathbf{z} = \mathbf{y} + (-1)^b c\mathbf{s}$$

with modulus $2q$ instead of q:



Note that M does not change if $\mathbf{y}$ is chosen from the uniform interval.

Bimodal rejection sampling

However, this makes "secure" implementation[3] much harder. It is basically due to "reject with probability a function of sk."

For e.g., for ≈120 bits security[4][5],

[3] an implementation secure against physical attacks (side-channel attacks)

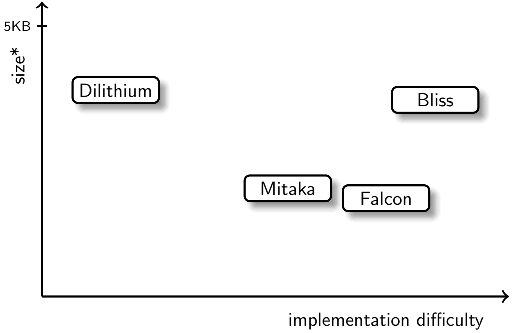
[4] core-SVP hardness

[5] size= |sig| + |vk|

Bimodal rejection sampling

However, this makes "secure" implementation[3] much harder. It is basically due to "reject with probability a function of sk."

For e.g., for ≈120 bits security[4][5],



[3] an implementation secure against physical attacks (side-channel attacks)
[4] core-SVP hardness
[5] size= $|\mathrm{sig}| + |\mathrm{vk}|$

HAETAE, in theory

The design rationale of HAETAE:

- **Fiat-Shamir with aborts** framework
- using **Bimodal** rejection sampling
- randomness sampling from **Hyperball** distribution

We now focus on **Hyperball** and the changes thereafter.

HAETAE, in theory

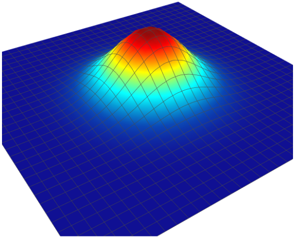
The design rationale of HAETAE:

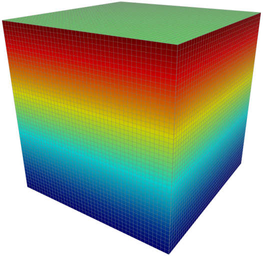
- **Fiat-Shamir with aborts** framework
- using **Bimodal** rejection sampling
- randomness sampling from **Hyperball** distribution

We now focus on **Hyperball** and the changes thereafter.

Hyperball bimodal rejection sampling

Previously, the randomness $\mathbf{y}$ was chosen from either discrete Gaussian or uniform hypercube[6].



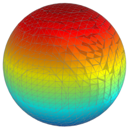


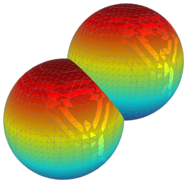
[6]The vectors $\mathbf{y}$ and $\mathbf{z}$ are high-dimensional vectors, so uniform in an interval is indeed a uniform hypercube.

Hyperball bimodal rejection sampling

We, instead, use **uniform hyperball** distribution for sampling $\mathbf{y}$ [DFPS22];

- to exploit optimal rejection rate,
- to reduce signature and verification key sizes,





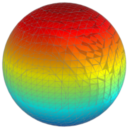
and use the **bimodal approach** [DDLL13];

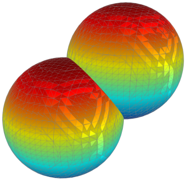
- for more compact signature sizes,
- for a simpler rejection condition, which leads to the easier implementation of secure rejection.

Hyperball bimodal rejection sampling

We, instead, use **uniform hyperball** distribution for sampling $\mathbf{y}$ [DFPS22];

- to exploit optimal rejection rate,
- to reduce signature and verification key sizes,



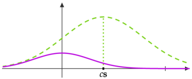


and use the **bimodal approach** [DDLL13];

- for more compact signature sizes,
- for a simpler rejection condition, which leads to the easier implementation of secure rejection.

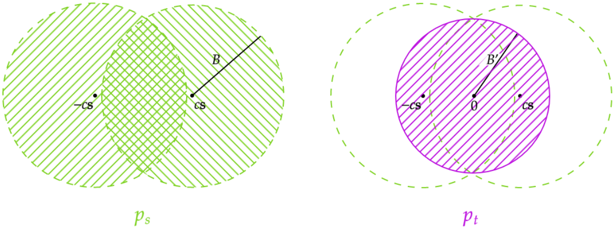
Hyperball bimodal rejection sampling

Recap: we return $\mathbf{x} = (c, \mathbf{z})$ with probability $\min\left(\frac{p_{\mathsf{t}}(\mathbf{x})}{M \cdot p_{\mathsf{s}}(\mathbf{x})}, 1\right)$.



We reject $\mathbf{x} = (c, \mathbf{z})$ sampled from a source distribution p_{s} to a target distribution p_{t}, where

- p_{s}: uniform in a hyperball of radii B centered at $\pm c\mathbf{s}$
  - union of two large balls
- p_{t}: uniform in a smaller hyperball of radii B' centered at zero
  - a smaller ball in the middle



Hyperball bimodal rejection sampling

Recap: we return $\mathbf{x} = (c, \mathbf{z})$ with probability $\min\left(\frac{p_{\mathsf{t}}(\mathbf{x})}{M \cdot p_{\mathsf{s}}(\mathbf{x})}, 1\right)$.

We reject $\mathbf{x} = (c, \mathbf{z})$ sampled from a source distribution p_{s} to a target distribution p_{t}, where

- $p_{\mathsf{s}}(\mathbf{x}) = \frac{1}{2 \cdot \mathsf{vol}(\mathcal{B}(B))} \cdot \chi_{\|\mathbf{z} - c\mathbf{s}\| < B} + \frac{1}{2 \cdot \mathsf{vol}(\mathcal{B}(B))} \cdot \chi_{\|\mathbf{z} + c\mathbf{s}\| < B}$,
- $p_{\mathsf{t}}(\mathbf{x}) = \frac{1}{\mathsf{vol}(\mathcal{B}(B))} \cdot \chi_{\|\mathbf{z}\| < B'}$.

This leads to

$$\frac{p_{\mathsf{t}}(\mathbf{x})}{M \cdot p_{\mathsf{s}}(\mathbf{x})} = \frac{\chi_{\|\mathbf{z}\| < B'}}{\chi_{\|\mathbf{z} - c\mathbf{s}\| < B} + \chi_{\|\mathbf{z} + c\mathbf{s}\| < B}}$$

$$= \begin{array}{ll} 0 & \text{if } \mathbf{z} \notin \mathcal{B}(B'), \\ 1/2 & \text{if } \mathbf{z} \in \mathcal{B}(B') \cap \mathcal{B}(B, c\mathbf{s}) \cap \mathcal{B}(B, -c\mathbf{s}), \\ 1 & \text{if } \mathbf{z} \in \mathcal{B}(B') \setminus (\mathcal{B}(B, c\mathbf{s}) \cap \mathcal{B}(B, -c\mathbf{s})) \end{array}$$

for some $M > 0$.

Hyperball bimodal rejection sampling

Recap: we return $\mathbf{x} = (c, \mathbf{z})$ with probability $\min\left(\frac{p_{\mathsf{t}}(\mathbf{x})}{M \cdot p_{\mathsf{s}}(\mathbf{x})}, 1\right)$.

We reject $\mathbf{x} = (c, \mathbf{z})$ sampled from a source distribution p_{s} to a target distribution p_{t}, where

- $p_{\mathsf{s}}(\mathbf{x}) = \frac{1}{2 \cdot \mathsf{vol}(\mathcal{B}(B))} \cdot \chi_{\|\mathbf{z}-c\mathbf{s}\|<B} + \frac{1}{2 \cdot \mathsf{vol}(\mathcal{B}(B))} \cdot \chi_{\|\mathbf{z}+c\mathbf{s}\|<B}$,
- $p_{\mathsf{t}}(\mathbf{x}) = \frac{1}{\mathsf{vol}(\mathcal{B}(B))} \cdot \chi_{\|\mathbf{z}\|<B'}$.

This leads to

$$\frac{p_{\mathsf{t}}(\mathbf{x})}{M \cdot p_{\mathsf{s}}(\mathbf{x})} = \frac{\chi_{\|\mathbf{z}\|<B'}}{\chi_{\|\mathbf{z}-c\mathbf{s}\|<B} + \chi_{\|\mathbf{z}+c\mathbf{s}\|<B}}$$

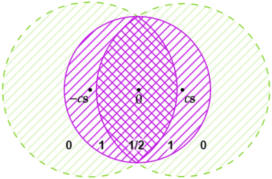
$$= \begin{cases} 0 & \text{if } \mathbf{z} \notin \mathcal{B}(B'), \\ 1/2 & \text{if } \mathbf{z} \in \mathcal{B}(B') \cap \mathcal{B}(B, c\mathbf{s}) \cap \mathcal{B}(B, -c\mathbf{s}), \\ 1 & \text{if } \mathbf{z} \in \mathcal{B}(B') \setminus (\mathcal{B}(B, c\mathbf{s}) \cap \mathcal{B}(B, -c\mathbf{s})) \end{cases}$$

for some $M > 0$.

Hyperball bimodal rejection sampling

That is, we return $\mathbf{x} = (c, \mathbf{z})$ with probability

- 0: if $\|\mathbf{z}\| \geq B'$,
- 1/2: else if $\|\mathbf{z} - c\mathbf{s}\| < B$ and $\|\mathbf{z} + c\mathbf{s}\| < B$,
- 1: otherwise.



Since $\mathbf{z} = \mathbf{y} + (-1)^b c\mathbf{s}$, we can do even simpler,

- if $\|\mathbf{z}\| \geq B'$, **reject**,
- else if $\|2\mathbf{z} - \mathbf{y}\| < B$,[7] **reject** with probability 1/2,
- otherwise, **accept**,

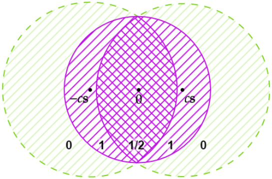
resulting in a signature, uniform in a hyperball $\mathcal{B}(B')$.

[7] $\{\mathbf{z} \pm c\mathbf{s}\} = \{\mathbf{y}, 2\mathbf{z} - \mathbf{y}\}$ and always $\|\mathbf{y}\| < B$.

Hyperball bimodal rejection sampling

That is, we return $\mathbf{x} = (c, \mathbf{z})$ with probability

- 0: if $\|\mathbf{z}\| \geq B'$,
- 1/2: else if $\|\mathbf{z} - c\mathbf{s}\| < B$ and $\|\mathbf{z} + c\mathbf{s}\| < B$,
- 1: otherwise.



Since $\mathbf{z} = \mathbf{y} + (-1)^b c\mathbf{s}$, we can do even simpler,

- if $\|\mathbf{z}\| \geq B'$, **reject**,
- else if $\|2\mathbf{z} - \mathbf{y}\| < B$,[7] **reject** with probability 1/2,
- otherwise, **accept**,

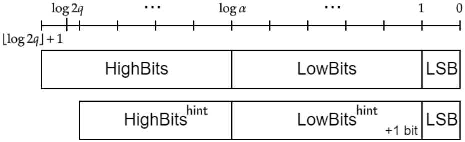
resulting in a signature, uniform in a hyperball $\mathcal{B}(B')$.

[7] $\{\mathbf{z} \pm c\mathbf{s}\} = \{\mathbf{y}, 2\mathbf{z} - \mathbf{y}\}$ and always $\|\mathbf{y}\| < B$.

Compression techniques

To reduce the size of the signature, we use two compression techniques:

- High, Low, and Least Significant Bits
  - basically, it is $\bmod^{\pm} \alpha$ for some power-of-two integer $\alpha \mid 2(q-1)$.
  - some optimizations for better sizes[8], e.g., HighBits$^{\text{hint}}$ and LowBits$^{\text{hint}}$: conserving one bit from HighBits while making LowBits a little bit complicated.



- Encoding via range Asymmetric Numeral System (rANS encoding)
  - rANS encoding is a type of entropy coding.
  - adapted from [Dud13], we encode high bits of signature within it's entropy $+1$ bit.

[8] newly updated!

Parameters and concrete security

Parameters sets Target security	HAETAE120 120	HAETAE180 180	HAETAE260 260
q (k, ℓ)	64513 (2,4)	64513 (3,6)	64513 (4,7)
Unforgeability (strong unforgeability for randomized version)			
Classical core-SVP Quantum core-SVP	123 (100) 108 (87)	189 (156) 166 (137)	258 (216) 227 (190)
Key security against key-recovery attack			
Classical core-SVP Quantum core-SVP	125 109	236 208	288 253
Sizes (in Bytes)			
$\|sig\|$ $\|vk\|$ $\|sig\| + \|vk\|$	1468 1056 2524	2285 1568 3853	2781 2080 4861

Table: Security and sizes for HAETAE.

Parameters and concrete security.

HAETAE has reasonable sizes and is easily implementable, and also seems securely maskable.

Targeting 120-bit security, we summarize recent lattice-based signatures. Sizes are shown in bytes. The prefixes *d* and *int* imply *discrete* and *integer*, respectively. Note that dHyperball requires continuous Gaussian at 0. Note that verification is fast enough in all the schemes.

Scheme	sig	vk	KeyGen	Sign	
				sampling	rejection
Dilithium-2	2420	1312	fast	Hypercube	$\lVert \cdot \rVert_\infty < B$
Bliss-1024[9]	1700	1792	fast	dGaussian at 0	reject with prob. $f(\mathsf{sk}, \mathsf{Sig})$
HAETAE120	1468	1056	fast	dHyperball at 0	$\lVert \cdot \rVert_2 < B$
Mitaka-512[10]	713	896	slow	dGaussian at 0 & intGaussian at $H(m)$	none
Falcon-512	666	897	slow	dGaussian at $H(m)$	none

Table: Comparison between different lattice-based signature schemes.

[9] modified Bliss (to ≥ 120 bit-security) in Dilithium paper.
[10] Mitaka-512 has 102 bits of security

Reference Implementation

Benchmark (CPU cycles and time elapsed)

- GNU/Linux with Linux kernel version 5.4.0.
- AMD Ryzen 3700x.
- The compiler gcc 9.4.0 with -O3 and -fomit-frame-pointer.

	HAETAE120	HAETAE180	HAETAE260
Keygen	730k	1,329k	1,867k
Sign	4,427k	6,843k	8,438k
Verify	491k	789k	1,145k
Total cycles	5,525k	8,961k	11,450k
Time elapsed	1.611ms	2.584ms	3.360ms

Table: Benchmark of current HAETAE (on-going)

Upcoming updates

Missing parts in the first round submission:

- rANS encoding
- rejection sampling for secret key
- min-entropy analysis

Modifications:

- Toward smaller sizes:
  - new compression and rANS encoding for hint
- Toward secure implementation:
  - get rid of floating-point arithmetic: numerical analysis and fixed-point Gaussian sampling is included for hyperball uniform sampling
- Toward faster implementation:
  - NTT/CRT-based implementation

The updated version will be uploaded to SMAUG & HAETAE website:
`http://kpqc.cryptolab.co.kr/`.

Thanks!

Any question?

References I

[Ajt96] M. Ajtai.
Generating hard instances of lattice problems (extended abstract).
In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, STOC '96, page 99–108, New York, NY, USA, 1996. Association for Computing Machinery.

[BG14] Shi Bai and Steven D Galbraith.
An improved compression technique for signatures based on learning with errors.
In Cryptographers' Track at the RSA Conference, pages 28–47. Springer, 2014.

[DDLL13] Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky.
Lattice signatures and bimodal gaussians.
In Annual Cryptology Conference, pages 40–56. Springer, 2013.

[DFPS22] Julien Devevey, Omar Fawzi, Alain Passelègue, and Damien Stehlé.
On rejection sampling in lyubashevsky's signature scheme.
Cryptology ePrint Archive, Number 2022/1249, 2022.
To be appeared in Asiacrypt, 2022. https://eprint.iacr.org/2022/1249.

References II

[DKL+18] Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé.
Crystals-dilithium: A lattice-based digital signature scheme.
IACR Transactions on Cryptographic Hardware and Embedded Systems, pages 238–268, 2018.

[DLP14] Léo Ducas, Vadim Lyubashevsky, and Thomas Prest.
Efficient identity-based encryption over ntru lattices.
In International Conference on the Theory and Application of Cryptology and Information Security, pages 22–41. Springer, 2014.

[DP16] Léo Ducas and Thomas Prest.
Fast fourier orthogonalization.
In Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation, pages 191–198, 2016.

[Duc14] Léo Ducas.
Accelerating bliss: the geometry of ternary polynomials.
Cryptology ePrint Archive, Paper 2014/874, 2014.
https://eprint.iacr.org/2014/874.

References III

[Dud13] Jarek Duda.
Asymmetric numeral systems: entropy coding combining speed of huffman coding with compression rate of arithmetic coding, 2013.
ArXiv preprint, available at `https://arxiv.org/abs/1311.2540`.

[EFG+22] Thomas Espitau, Pierre-Alain Fouque, François Gérard, Mélissa Rossi, Akira Takahashi, Mehdi Tibouchi, Alexandre Wallet, and Yang Yu.
Mitaka: A simpler, parallelizable, maskable variant of.
In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pages 222–253. Springer, 2022.

[ETWY22] Thomas Espitau, Mehdi Tibouchi, Alexandre Wallet, and Yang Yu.
Shorter hash-and-sign lattice-based signatures.
In Yevgeniy Dodis and Thomas Shrimpton, editors, Advances in Cryptology – CRYPTO, 2022.

[FHK+18] Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang.
Falcon: Fast-fourier lattice-based compact signatures over ntru.
Submission to the NIST's post-quantum cryptography standardization process, 36(5), 2018.

References IV

[GLP12] Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann.
Practical lattice-based cryptography: A signature scheme for embedded systems.
In International Workshop on Cryptographic Hardware and Embedded Systems, pages 530–547. Springer, 2012.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan.
Trapdoors for hard lattices and new cryptographic constructions.
In Proceedings of the fortieth annual ACM symposium on Theory of computing, pages 197–206, 2008.

[HHGP+03] Jeffrey Hoffstein, Nick Howgrave-Graham, Jill Pipher, Joseph H Silverman, and William Whyte.
Ntrusign: Digital signatures using the ntru lattice.
In Cryptographers' track at the RSA conference, pages 122–140. Springer, 2003.

[HR07] Ishay Haviv and Oded Regev.
Tensor-based hardness of the shortest vector problem to within almost polynomial factors.
In Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing, STOC '07, page 469–477, New York, NY, USA, 2007. Association for Computing Machinery.

References V

[KLS18] Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner.
A concrete treatment of Fiat-Shamir signatures in the quantum random-oracle model.
In Advances in Cryptology – EUROCRYPT, pages 552–586. Springer, 2018.

[Lyu09] Vadim Lyubashevsky.
Fiat-shamir with aborts: Applications to lattice and factoring-based signatures.
In International Conference on the Theory and Application of Cryptology and Information Security, pages 598–616. Springer, 2009.

[Lyu12] Vadim Lyubashevsky.
Lattice signatures without trapdoors.
In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pages 738–755. Springer, 2012.

HAETAE description (high-level)

KeyGen(1^λ)

1: $\mathbf{A}_{\text{gen}} \leftarrow \mathcal{R}_q^{k\times(\ell-1)}$ and $(\mathbf{s}_{\text{gen}}, \mathbf{e}_{\text{gen}}) \leftarrow S_\eta^{\ell-1} \times S_\eta^k$
2: $\mathbf{b} = \mathbf{A}_{\text{gen}} \cdot \mathbf{s}_{\text{gen}} + \mathbf{e}_{\text{gen}} \in \mathcal{R}_q^k$
3: $\mathbf{A} = (-2\mathbf{b} + q\mathbf{j} \mid 2\mathbf{A}_{\text{gen}} \mid 2\mathbf{Id}_k) \bmod 2q$ and write $\mathbf{A} = (\mathbf{A}_1 \mid 2\mathbf{Id}_k)$
4: $\mathbf{s} = (1, \mathbf{s}_{\text{gen}}, \mathbf{e}_{\text{gen}})$
5: if $\sigma_{\max}(\mathrm{rot}(\mathbf{s}_{\text{gen}})) > \gamma$, then restart
6: Return $\mathsf{sk} = \mathbf{s}$, $\mathsf{vk} = \mathbf{A}$

Sign(sk, M)

1: $\mathbf{y} \leftarrow U(\mathcal{B}_{(1/N)\mathcal{R},(k+\ell)}(B))$
2: $c = H(\mathrm{HighBits}_{2q}^{\text{hint}}(\mathbf{A}\lfloor \mathbf{y} \rceil, \alpha), \mathrm{LSB}(\lfloor y_0 \rceil), M) \in \mathcal{R}_2$
3: $\mathbf{z} = (\mathbf{z}_1, \mathbf{z}_2) = \mathbf{y} + (-1)^b c \cdot \mathbf{s}$ for $b \leftarrow U(\{0,1\})$
4: $\mathbf{h} = \mathrm{HighBits}_{2q}^{\text{hint}}(\mathbf{A}\lfloor \mathbf{z} \rceil - qc\mathbf{j}, \alpha) - \mathrm{HighBits}_{2q}^{\text{hint}}(\mathbf{A}_1 \lfloor \mathbf{z}_1 \rceil - qc\mathbf{j}, \alpha) \bmod^+ \frac{2(q-1)}{\alpha}$
5: if $\|\mathbf{z}\|_2 \geq B'$, then restart
6: if $\|2\mathbf{z} - \mathbf{y}\|_2 < B$, then restart with probability $1/2$
7: Return $\sigma = (\mathrm{Encode}(\mathrm{HighBits}(\lfloor \mathbf{z}_1 \rceil, a)), \mathrm{LowBits}(\lfloor \mathbf{z}_1 \rceil, a), \mathrm{Encode}(\mathbf{h}), c)$

Verify(vk, M, $\sigma = (x, \mathbf{v}, h, c)$)

1: $\tilde{\mathbf{z}}_1 = \mathrm{Decode}(x) \cdot a + \mathbf{v}$ and $\tilde{\mathbf{h}} = \mathrm{Decode}(h)$
2: $\mathbf{w} = \tilde{\mathbf{h}} + \mathrm{HighBits}_{2q}^{\text{hint}}(\mathbf{A}_1 \tilde{\mathbf{z}}_1 - qc\mathbf{j}, \alpha) \bmod^+ \frac{2(q-1)}{\alpha}$
3: $w' = \mathrm{LSB}(\tilde{z}_0 - c)$
4: $\tilde{\mathbf{z}}_2 = \left[\mathbf{w} \cdot \alpha + w'\mathbf{j} - (\mathbf{A}_1 \tilde{z}_1 - qc\mathbf{j})\right]/2 \bmod^{\pm} q$
5: $\tilde{\mathbf{z}} = (\tilde{\mathbf{z}}_1, \tilde{\mathbf{z}}_2)$
6: Return $(\ c = H(\mathbf{w}, w', M)\ ) \wedge (\ \|\tilde{\mathbf{z}}\| < B''\ )$