Lattice-Based Signatures

Thomas Prest

PQShield (UK/FR/NL/...)

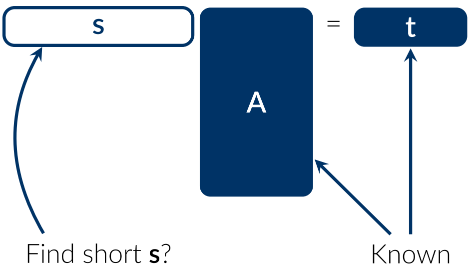
PQCifris 2022 Workshop

https://tprest.github.io/pdf/slides/pqcifris-2022.pdf

Introduction

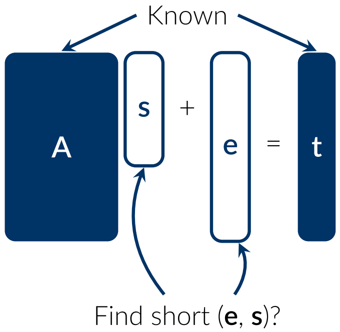
Lattice assumptions

Short integer solution (SIS)



- Used in dense/surjective regime
- Gets easier when $\|\mathbf{s}\|$ increases
- Also hard to solve approximately [CGM19]

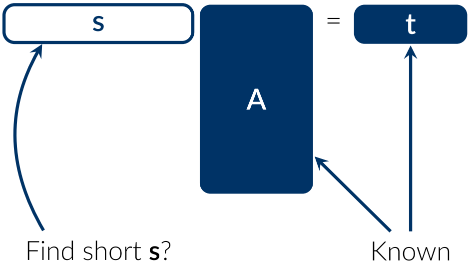
Learning with errors (LWE)



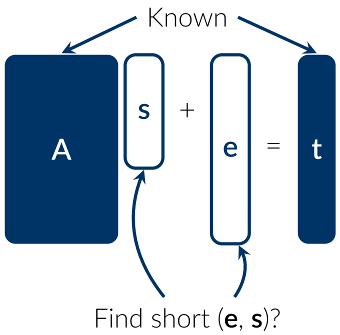
- Used in sparse/injective regime
- Gets harder when $\|(\mathbf{s}, \mathbf{e})\|$ increases
- Also hard to solve approximately

Lattice assumptions

Short integer solution (SIS)



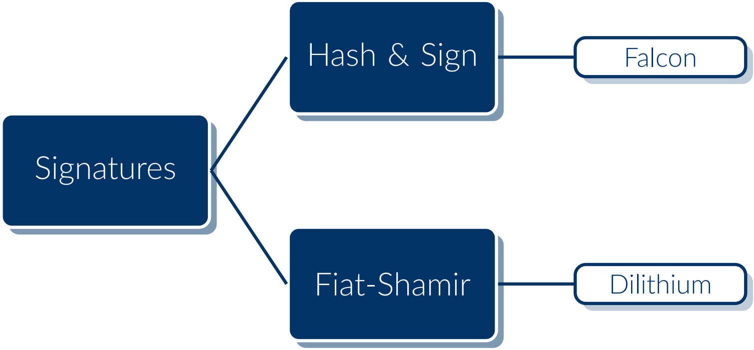
Learning with errors (LWE)



NTRU

Given $h \in \mathcal{R}_q = \mathbb{Z}_q[x]/(\varphi)$, find small f, g such that $g \cdot f^{-1} = h$ (i.e. $\begin{bmatrix} f & g \end{bmatrix} \cdot \begin{bmatrix} h \\ -1 \end{bmatrix} = \mathbf{0}$)

Roadmap



Part I: Hash & Sign

- High-level principle
- Choice of lattice class
- Choice of sampler

Part II: Fiat-Shamir

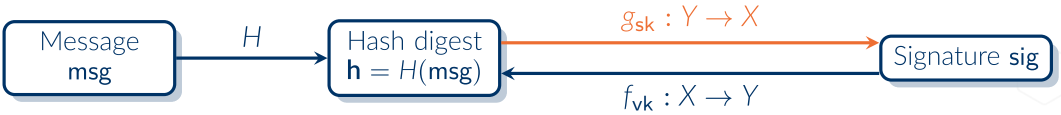
- High-level principle
- Choice of lattice class
- Ninja tricks
- Choice of distribution

Warning:

- Some mathematics are oversimplified
- Does not cover recent schemes based on the lattice isomorphism problem (LIP)

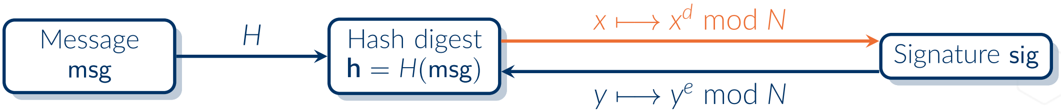
Hash-then-Sign

Signatures based on *Hash-then-Sign*



- **The signer** computes $\mathbf{h} = H(\mathsf{msg})$, then $\mathsf{sig} = g_{\mathsf{sk}}(\mathbf{h})$ using the signing key **sk**.
- **The verifier** computes $\mathbf{h} = H(\mathsf{msg})$, then $\mathbf{h}' = f_{\mathsf{vk}}(\mathsf{sig})$ using the verification key **vk**, and checks that the results match (i.e. $\mathbf{h}' = \mathbf{h}$).

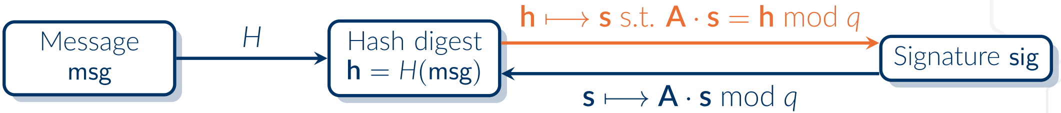
The case of RSA signatures



Example with RSA signatures:

- $g_{\mathbf{sk}}(x) = x^d \bmod N$, and $f_{\mathbf{vk}}(y) = y^e \bmod N$.
- $e \cdot d = 1 \bmod \phi(N)$

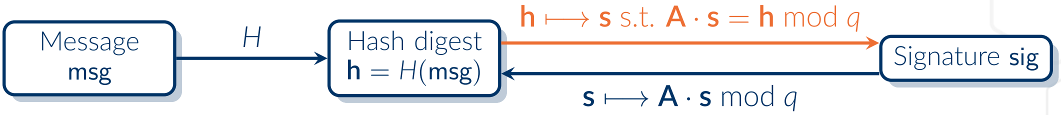
The case of lattices (first attempt)



First attempt with lattice (not secure):

- *Verification key:* **vk** is a (pseudo)random matrix $\mathbf{A} \in \mathcal{R}_q^{n \times m}$.
- *Signing key:* **sk** is a short matrix $\mathbf{B} \in \mathcal{R}_q^{m \times m}$ such that $\mathbf{A} \cdot \mathbf{B} = \mathbf{0} \bmod q$.
- *Signing:*
  1. Hash **msg** to a point $\mathbf{h} \in \mathcal{R}_q^n$.
  2. Compute $\mathbf{c} \in \mathcal{R}_q^m$ s.t. $\mathbf{A} \cdot \mathbf{c} = \mathbf{h}$.
  3. Compute $\mathbf{v} \in \mathbf{B} \cdot \mathcal{R}_q^m$ close to $\mathbf{v}$ (the hard part, see next slide)
  4. The signature is $\mathbf{s} := \mathbf{c} - \mathbf{v}$
- *Verification:*
  1. Check that $\mathbf{A} \cdot \mathbf{s} = \mathbf{h}$.
  2. Check that $\mathbf{s}$ is short (say, $\|\mathbf{s}\|_2$ is small).

The case of lattices (first attempt)



First attempt with lattice (not secure):

- *Verification key:* vk is a (pseudo)random matrix $\mathbf{A} \in \mathcal{R}_q^{n \times m}$.
- *Signing key:* sk is a short matrix $\mathbf{B} \in \mathcal{R}_q^{m \times m}$ such that $\mathbf{A} \cdot \mathbf{B} = \mathbf{0} \bmod q$.
- *Signing:*
  1. Hash msg to a point $\mathbf{h} \in \mathcal{R}_q^n$.
  2. Compute $\mathbf{c} \in \mathcal{R}_q^m$ s.t. $\mathbf{A} \cdot \mathbf{c} = \mathbf{h}$.
  3. Compute $\mathbf{v} \in \mathbf{B} \cdot \mathcal{R}_q^m$ close to $\mathbf{v}$ (the hard part, see next slide)
  4. The signature is $\mathbf{s} := \mathbf{c} - \mathbf{v}$
- *Verification:*
  1. Check that $\mathbf{A} \cdot \mathbf{s} = \mathbf{h}$.
  2. Check that $\mathbf{s}$ is short (say, $\|\mathbf{s}\|_2$ is small).

Big questions:

- How do we generate a suitable keypair $(\mathbf{A}, \mathbf{B})$?
- How do we compute $\mathbf{v}$ close to $\mathbf{v}$?

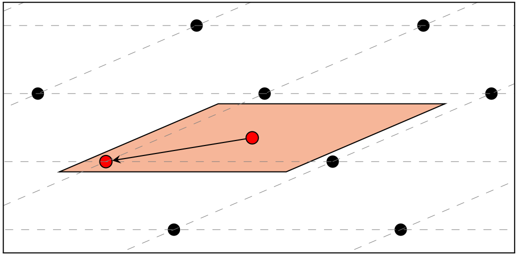
Computing a lattice point v close to the target c

For NearestPlane, the Gram-Schmidt orthogonalization $\mathbf{B} = \mathbf{L} \cdot \tilde{\mathbf{B}}$ is precomputed.

RoundOff(B, c)

1. $\mathbf{t} \leftarrow \mathbf{c} \cdot \mathbf{B}^{-1}$
2. For $j \in \{n, \ldots, 1\}$:
   1. $z_j \leftarrow \lceil t_j \rfloor$
3. Return $\mathbf{v} := \mathbf{z} \cdot \mathbf{B}$

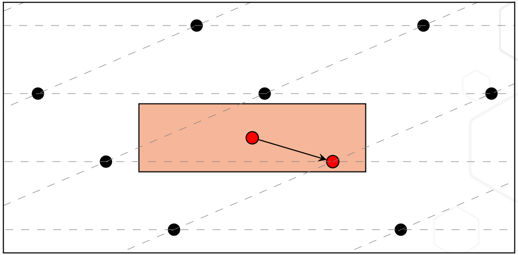
$\Downarrow$



NearestPlane(B, L, c)

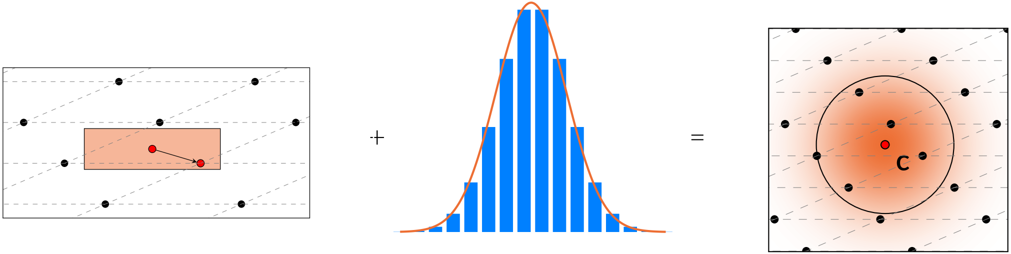
1. $\mathbf{t} \leftarrow \mathbf{c} \cdot \mathbf{B}^{-1}$
2. For $j \in \{n, \ldots, 1\}$:
   1. $z_j \leftarrow \lceil t_j + \sum_{i>j} (t_1 - z_i) L_{i,j} \rfloor$
3. Return $\mathbf{v} := \mathbf{z} \cdot \mathbf{B}$

$\Downarrow$



Trapdoor sampling [GPV08]

Problem: the distribution of signatures may leak the shape of **B**
Solution: randomize the solving procedure with Gaussians



Computing a suitable (A, B) – NTRU trapdoors

NTRU trapdoors

Let $f, g, F, G \in \mathcal{R}$ such that:

$$fG - gF = q \tag{1}$$

$$h := g/f \bmod q \tag{2}$$

We set $\mathbf{A} = \begin{bmatrix} 1 & h \end{bmatrix}$ and $\mathbf{B} = \begin{bmatrix} g & G \\ -f & -F \end{bmatrix}$.

Computing a suitable (A, B) – NTRU trapdoors

NTRU trapdoors

Let $f, g, F, G \in \mathcal{R}$ such that:

$$fG - gF = q \tag{1}$$

$$h := g/f \bmod q \tag{2}$$

We set $\mathbf{A} = \begin{bmatrix} 1 & h \end{bmatrix}$ and $\mathbf{B} = \begin{bmatrix} g & G \\ -f & -F \end{bmatrix}$.

- **Pseudorandomness of A:** NTRU assumption.
- **Orthogonality:** One can easily show that $\mathbf{A} \cdot \mathbf{B} = \mathbf{0} \bmod q$.
- **Shortness of B:** Given (f, g), one can compute suitable (F, G) such that

$$\|(F, G)\| \approx \underbrace{\frac{q}{\|(f, g)\|}}_{\text{component} \perp (f,g)} + \underbrace{\sqrt{\frac{d}{12}} \cdot \|(f, g)\|}_{\text{component} \parallel (f,g)} \tag{3}$$

Computing a suitable (A, B) – [MP12] trapdoors

Gadget matrices

We define $\mathbf{g}, \mathbf{B}$ such that $\mathbf{g} \cdot \mathbf{B} = \mathbf{0} \bmod q$:

→ $$\mathbf{g} = (1, b, b^2, \ldots, b^{k-1}) \text{ and } \mathbf{B} = \begin{bmatrix} b & & & q_0 \\ -1 & \ddots & & \vdots \\ & \ddots & b & q_{k-2} \\ & & -1 & q_{k-1} \end{bmatrix}, \text{ where } q = \sum_i q_i b^i$$

→ The "gadget matrix" $\mathbf{G} = \mathbf{I}_n \otimes \mathbf{g}$ and $\mathbf{G}^\perp = \mathbf{I}_n \otimes \mathbf{B}$ also satisfy $\mathbf{G} \cdot \mathbf{G}^\perp = \mathbf{0} \bmod q$.

Generating a Micciancio-Peikert trapdoor

→ Set $\bar{\mathbf{A}} = \begin{bmatrix} \hat{\mathbf{A}} & \mathbf{I} \end{bmatrix}$, where $\hat{\mathbf{A}}$ is a uniformly random matrix.

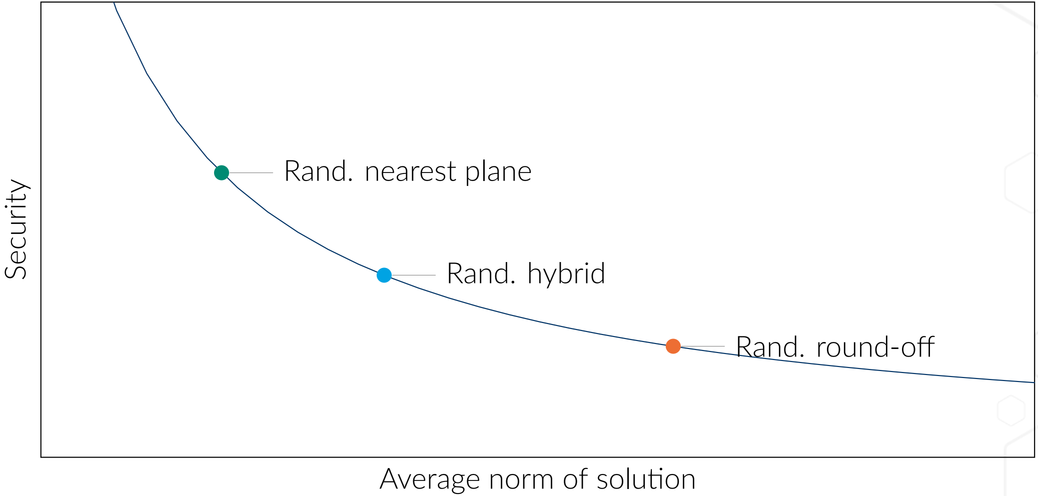
→ Generate a short random matrix $\mathbf{R}$

→ Set $\mathbf{A} = [\bar{\mathbf{A}} \quad \mathbf{G} - \bar{\mathbf{A}} \cdot \mathbf{R}]$ and $\mathbf{B} = \begin{bmatrix} \mathbf{R} \\ \mathbf{I} \end{bmatrix} \cdot \mathbf{G}^\perp$.

Pseudorandomness (under LWE), orthonogality and shortness: Exercise.

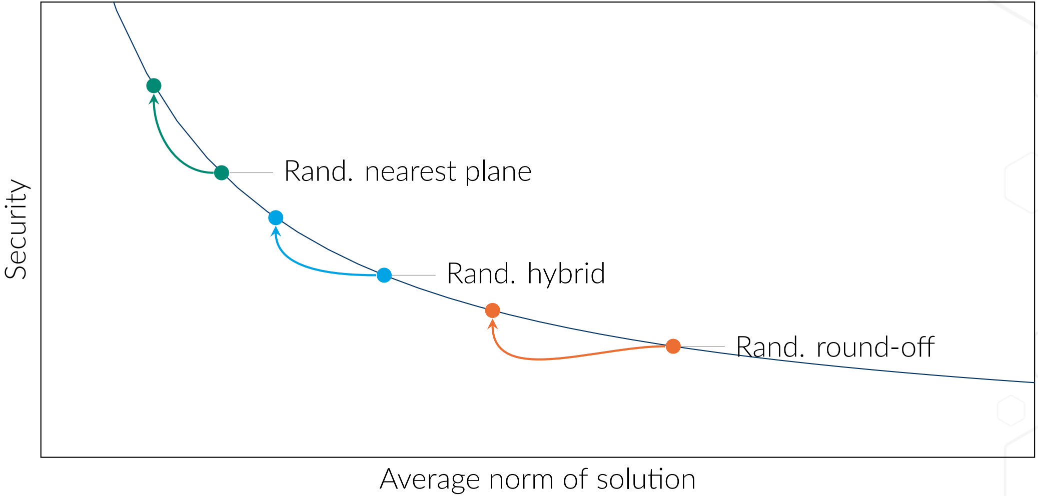
Example with NTRU trapdoors

Remember SIS (solving $\mathbf{A} \cdot \mathbf{s} = \mathbf{t}$) gets harder when $||\mathbf{s}||$ is shorter.



Example with NTRU trapdoors

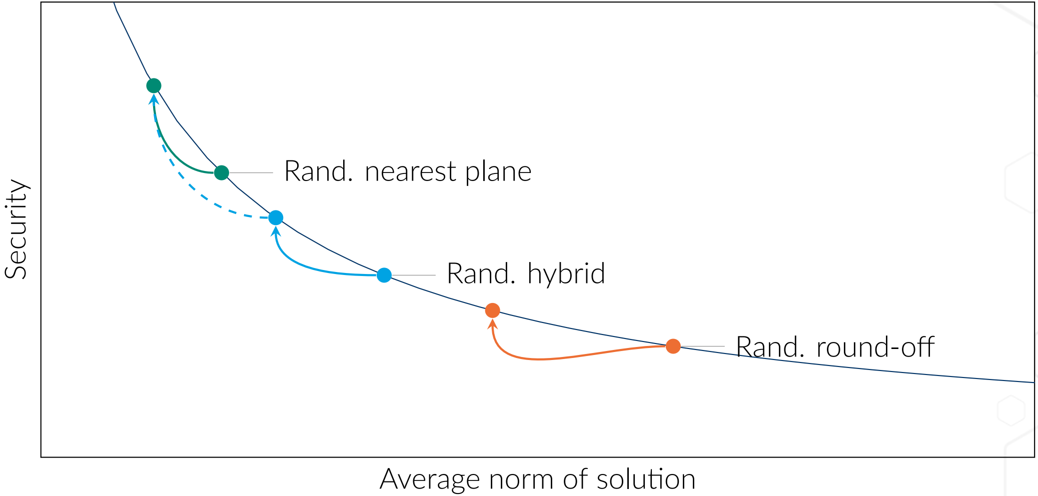
Remember SIS (solving $\mathbf{A} \cdot \mathbf{s} = \mathbf{t}$) gets harder when $\|\mathbf{s}\|$ is shorter.



→ 2017: Improved statistical analyses w/ R'enyi divergence (solid arrows)

Example with NTRU trapdoors

Remember SIS (solving $\mathbf{A} \cdot \mathbf{s} = \mathbf{t}$) gets harder when $\|\mathbf{s}\|$ is shorter.



- 2017: Improved statistical analyses w/ R'enyi divergence (solid arrows)
- 2022: Improved generation of NTRU trapdoors (dashed arrow)

Section conclusion and bibliography

Foundations

- Trapdoor sampling [GPV08]
- Micciancio-Peikert sampling [MP12]

Trapdoor samplers

- Randomised nearest plane [GPV08]
- Randomised round-off [Pei10]
- Hybrid [Pre15]
- Fast Fourier sampling [DP16]

Trapdoor lattices

- NTRU lattices [HHP+03, DLP14]
- Micciancio-Peikert trapdoors [MP12, CGM19]
- Improved NTRU trapdoors [ea22]

Efficient instantiations

- Falcon (NTRU) [PFH+17]
- Mitaka (NTRU) [EFG+22]
- (Micciancio-Peikert) [CGM19]

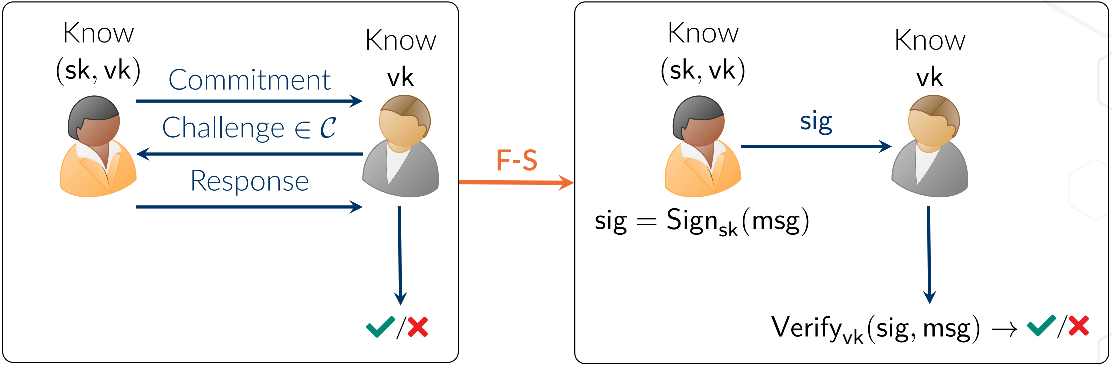
Proof techniques

- Security model [GPV08, CGM19]
- Statistical relaxations [Pre17]

Fiat-Shamir Signatures

Fiat–Shamir Signatures

(3-Move) Identification Protocol — **Signature Scheme**

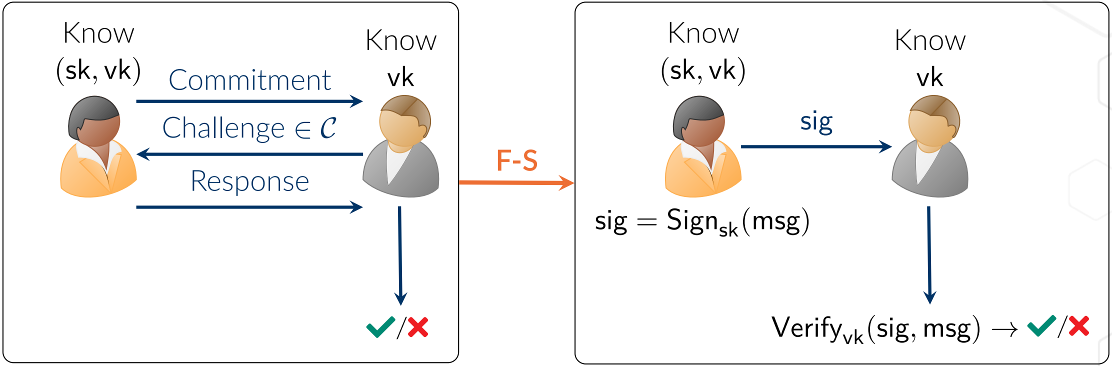


F-S refers to the Fiat-Shamir transform:

- → The challenge is now defined as $H(\text{Commitment}\|\textbf{msg})$.
- → The signature is (Commitment,Response).

Fiat–Shamir Signatures

(3-Move) Identification Protocol | **Signature Scheme**



We obtain an existentially unforgeable signature scheme in the ROM if the ID protocol is:

1. **Correct:** An honest prover can convince a verifier he knows sk
2. **Honest verifier zero-knowledge:** A valid transcript can be simulated without sk
3. **Soundness:** A dishonest prover cannot convince a verifier he knows sk

Schnorr signatures (Fiat-Shamir w/ discrete log)

Keygen($g \in \mathbb{G}$)

1. $x \leftarrow \mathbb{Z}_q^\times$ $\quad (q = |\mathbb{G}|)$
2. $h \leftarrow g^x$
3. $\mathsf{sk} := x, \mathsf{vk} := h$

Sign(msg, sk)

1. $r \leftarrow \mathbb{Z}_q^\times$
2. $u \leftarrow g^r$ (Commitment)
3. $c \leftarrow H(u \| \mathsf{msg})$ (Challenge)
4. $z \leftarrow r - cx$ (Response)
5. $\mathsf{sig} := (u, z)$

Verify(msg, vk)

1. Accept if and only if $(g^z \cdot h^c = u)$

It is easy to show:

- ✓ **Correctness**
- ✓ **HVZK**
- ✓ **Special soundness**

Note that **DSA** and **ECDSA** are very similar to this scheme.

Fiat-Shamir with SIS

Keygen($\mathbf{A} \in \mathcal{R}_q^{k \times \ell}$)

1. $\mathbf{s} \leftarrow \chi_1$ (short)
2. $\mathbf{t} \leftarrow \mathbf{A}\mathbf{s}$
3. $\mathsf{sk} := \mathbf{s}, \mathsf{vk} := \mathbf{t}$

Sign(msg, sk)

1. $\mathbf{r} \leftarrow \chi_2$ (short)
2. $\mathbf{u} \leftarrow \mathbf{A}\mathbf{r}$
3. $\mathbf{c} \leftarrow H(\mathbf{u} \| \mathsf{msg})$
4. $\mathbf{z} \leftarrow \mathbf{r} - \mathbf{c}\mathbf{s}$
5. $\mathsf{sig} := (\mathbf{u}, \mathbf{z})$

Verify(msg, vk, sig)

1. Accept iff ($\mathbf{z}$ is short) and ($\mathbf{A}\mathbf{z} - \mathbf{c}\mathbf{t} = \mathbf{u}$).

Fiat-Shamir with SIS

Keygen($\mathbf{A} \in \mathcal{R}_q^{k \times \ell}$)

1. $\mathbf{s} \leftarrow \chi_1$ (short)
2. $\mathbf{t} \leftarrow \mathbf{A}\mathbf{s}$
3. $\mathsf{sk} := \mathbf{s}, \mathsf{vk} := \mathbf{t}$

Sign(msg, sk)

1. $\mathbf{r} \leftarrow \chi_2$ (short)
2. $\mathbf{u} \leftarrow \mathbf{A}\mathbf{r}$
3. $\mathbf{c} \leftarrow H(\mathbf{u} \| \mathsf{msg})$
4. $\mathbf{z} \leftarrow \mathbf{r} - \mathbf{c}\mathbf{s}$
5. $\mathsf{sig} := (\mathbf{u}, \mathbf{z})$

Verify(msg, vk, sig)

1. Accept iff ($\mathbf{z}$ is short) and ($\mathbf{A}\mathbf{z} - \mathbf{c}\mathbf{t} = \mathbf{u}$).

- ✓ Correctness
- ✗ HVZK
- ✗ Special soundness

Fiat-Shamir with SIS

Keygen$(\mathbf{A} \in \mathcal{R}_q^{k \times \ell})$

1. $\mathbf{s} \leftarrow \chi_1$ (short)
2. $\mathbf{t} \leftarrow \mathbf{As}$
3. $\mathsf{sk} := \mathbf{s}, \mathsf{vk} := \mathbf{t}$

Sign$(\mathsf{msg}, \mathsf{sk})$

1. $\mathbf{r} \leftarrow \chi_2$ (short)
2. $\mathbf{u} \leftarrow \mathbf{Ar}$
3. $\mathbf{c} \leftarrow H(\mathbf{u} \| \mathsf{msg})$ (short)
4. $\mathbf{z} \leftarrow \mathbf{r} - \mathbf{cs}$
5. $\mathsf{sig} := (\mathbf{u}, \mathbf{z})$

Verify$(\mathsf{msg}, \mathsf{vk}, \mathsf{sig})$

1. Accept iff ($\mathbf{z}$ is short) and ($\mathbf{Az} - \mathbf{ct} = \mathbf{u}$).

Soundness: Using rewinding:

- Transcript 1: $(\mathbf{u}, \mathbf{c}, \mathbf{z} \mid \mathbf{Az} - \mathbf{ct} = \mathbf{u})$
- Transcript 2: $(\mathbf{u}, \mathbf{c}', \mathbf{z}' \mid \mathbf{Az}' - \mathbf{c}'\mathbf{t} = \mathbf{u})$

$$[\,\mathbf{A} \,\|\, \mathbf{t}\,] \cdot \begin{bmatrix} \mathbf{z} - \mathbf{z}' \\ \mathbf{c} - \mathbf{c}' \end{bmatrix} = \mathbf{0} \qquad (4)$$

- ✓ **Correctness**
- ✗ **HVZK**
- ✓ **Special soundness** (imperfect) is satisfied, as long as $\mathbf{c}$ is short.

Fiat-Shamir with SIS

Keygen($\mathbf{A} \in \mathcal{R}_q^{k \times \ell}$)

1. $\mathbf{s} \leftarrow \chi_1$ (short)
2. $\mathbf{t} \leftarrow \mathbf{A}\mathbf{s}$
3. $\mathrm{sk} := \mathbf{s}, \mathrm{vk} := \mathbf{t}$

Sign(msg, sk)

1. $\mathbf{r} \leftarrow \chi_2$ (short)
2. $\mathbf{u} \leftarrow \mathbf{A}\mathbf{r}$
3. $\mathbf{c} \leftarrow H(\mathbf{u} \| \mathrm{msg})$ (short)
4. $\mathbf{z} \leftarrow \mathbf{r} - \mathbf{c}\mathbf{s}$
5. Rejection sampling step
6. $\mathrm{sig} := (\mathbf{u}, \mathbf{z})$

Verify(msg, vk, sig)

1. Accept iff ($\mathbf{z}$ is short) and ($\mathbf{A}\mathbf{z} - \mathbf{c}\mathbf{t} = \mathbf{u}$).

- ✓ **Correctness**
- ✓ **HVZK** requires rejection sampling.
- ✓ **Special soundness** (imperfect) is satisfied, as long as $\mathbf{c}$ is short.

Without rejection sampling, statistical attacks may recover the signing key.

Fiat-Shamir with SIS

Keygen($\mathbf{A} \in \mathcal{R}_q^{k \times \ell}$)

1. $\mathbf{s} \leftarrow \chi_1$ (short)
2. $\mathbf{t} \leftarrow \mathbf{A}\mathbf{s}$
3. $\mathsf{sk} := \mathbf{s}, \mathsf{vk} := \mathbf{t}$

Sign(msg, sk)

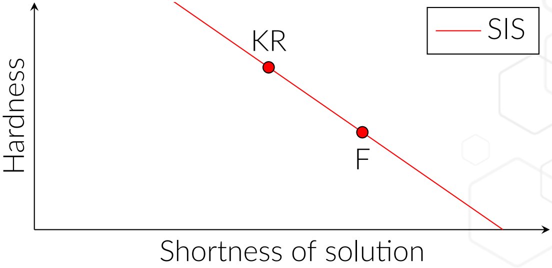
1. $\mathbf{r} \leftarrow \chi_2$ (short)
2. $\mathbf{u} \leftarrow \mathbf{A}\mathbf{r}$
3. $\mathbf{c} \leftarrow H(\mathbf{u} \| \mathsf{msg})$ (short)
4. $\mathbf{z} \leftarrow \mathbf{r} - \mathbf{c}\mathbf{s}$
5. Rejection sampling step
6. $\mathsf{sig} := (\mathbf{u}, \mathbf{z})$

Verify(msg, vk, sig)

1. Accept iff ($\mathbf{z}$ is short) and ($\mathbf{A}\mathbf{z} - \mathbf{c}\mathbf{t} = \mathbf{u}$).

Concrete hardness:

- *Key-recovery:* SIS with a short $\mathbf{s}$
- *Forgery:* SIS with a short-ish $\mathbf{z}$



Fiat-Shamir w/ (LWE+SIS) [Lyu12]

Keygen($\mathbf{A} \in \mathcal{R}_q^{k \times \ell}$)

1. $\mathbf{s}_1, \mathbf{s}_2 \leftarrow \chi_1 \times \chi_2$ (short)
2. $\mathbf{t} \leftarrow \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$
3. $\mathsf{sk} := (\mathbf{s}_1, \mathbf{s}_2), \mathsf{vk} := \mathbf{t}$

Sign(msg, sk)

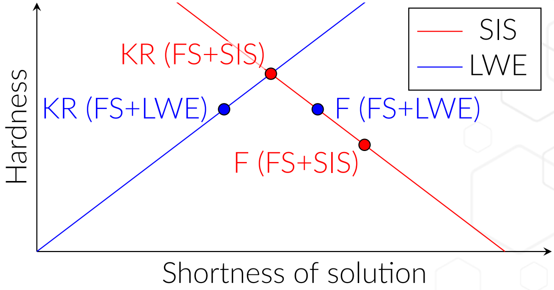
1. $\mathbf{r}_1, \mathbf{r}_2 \leftarrow \chi_3 \times \chi_4$ (short)
2. $\mathbf{u} \leftarrow \mathbf{A}\mathbf{r}_1 + \mathbf{r}_2$
3. $\mathbf{c} \leftarrow H(\mathbf{u} \| \mathsf{msg})$ (short)
4. $\mathbf{z}_1 \leftarrow \mathbf{r}_1 - \mathbf{c}\mathbf{s}_1$
5. $\mathbf{z}_2 \leftarrow \mathbf{r}_2 - \mathbf{c}\mathbf{s}_2$
6. Rejection sampling step
7. $\mathsf{sig} := (\mathbf{u}, \mathbf{z}_1, \mathbf{z}_2)$

Verify(msg, vk, sig)

1. Accept iff $(\mathbf{z}_1, \mathbf{z}_2)$ is short and $\mathbf{A}\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{t}\mathbf{c} = \mathbf{u}$

Concrete hardness:

- *Key-recovery:* LWE with a short $\mathbf{s}$
- *Forgery:* SIS with a short-ish $\mathbf{z}$



Fiat-Shamir w/ (LWE+SIS) [Lyu12]

Keygen($\mathbf{A} \in \mathcal{R}_q^{k\times\ell}$)

1. $\mathbf{s}_1, \mathbf{s}_2 \leftarrow \chi_1 \times \chi_2$ (short)
2. $\mathbf{t} \leftarrow \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$
3. $\mathsf{sk} := (\mathbf{s}_1, \mathbf{s}_2), \mathsf{vk} := \mathbf{t}$

Sign(msg, sk)

1. $\mathbf{r}_1, \mathbf{r}_2 \leftarrow \chi_3 \times \chi_4$ (short)
2. $\mathbf{u} \leftarrow \mathbf{A}\mathbf{r}_1 + \mathbf{r}_2$
3. $\mathbf{c} \leftarrow H(\mathbf{u} \| \mathsf{msg})$ (short)
4. $\mathbf{z}_1 \leftarrow \mathbf{r}_1 - \mathbf{c}\mathbf{s}_1$
5. $\mathbf{z}_2 \leftarrow \mathbf{r}_2 - \mathbf{c}\mathbf{s}_2$
6. Rejection sampling step
7. $\mathsf{sig} := (\mathbf{u}, \mathbf{z}_1, \mathbf{z}_2)$

Verify(msg, vk, sig)

1. Accept iff $(\mathbf{z}_1, \mathbf{z}_2)$ is short and $\mathbf{A}\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{t}\mathbf{c} = \mathbf{u}$

LWE also allows **two** optimisations that can be summarised by:

> "*If you are solving LWE for* $(\mathbf{A}, \mathbf{t} + \mathbf{e})$,
> *you are also solving LWE for* $(\mathbf{A}, \mathbf{t})$."

We will note MSB := "most significant bits" (the proportion may vary).

Fiat-Shamir w/ (LWE+SIS) - Optimisation 1

PQShield

Keygen($\mathbf{A} \in \mathcal{R}_q^{k\times\ell}$)

1. $\mathbf{s}_1, \mathbf{s}_2 \leftarrow \chi_1 \times \chi_2$ (short)
2. $\mathbf{t} \leftarrow \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$
3. $\mathsf{sk} := (\mathbf{s}_1, \mathbf{s}_2), \mathsf{vk} := \mathbf{t}$

Sign(msg, sk)

1. $\mathbf{r} \leftarrow \chi_3$ (short)
2. $\mathbf{u} \leftarrow \mathsf{MSB}(\mathbf{A}\mathbf{r})$
3. $\mathbf{c} \leftarrow H(\mathbf{u} \| \mathsf{msg})$ (short)
4. $\mathbf{z} \leftarrow \mathbf{r} - \mathbf{c}\mathbf{s}_1$
5. Rejection sampling step
6. $\mathsf{sig} := (\mathbf{u}, \mathbf{z})$

Verify(msg, vk, sig)

1. Accept iff $\mathbf{z}$ is short and $\mathsf{MSB}(\mathbf{Az}\text{- }\mathbf{tc}) = \mathbf{u}$

Bai-Galbraith trick [BG14]: the response sends only $\mathbf{z} := \mathbf{z}_1$ instead of $(\mathbf{z}_1, \mathbf{z}_2)$.

- → To preserve correctness, only check that $(\mathbf{Az} - \mathbf{tc})$ and $\mathbf{u}$ match on their MSBs.
- → If moderate, bit dropping only mildly affect the hardness of LWE.

Fiat-Shamir w/ (LWE+SIS) - Optimisation 2

PQSHIELD

Keygen($\mathbf{A} \in \mathcal{R}_q^{k \times \ell}$)

1. $\mathbf{s}_1, \mathbf{s}_2 \leftarrow \chi_1 \times \chi_2$ (short)
2. $\mathbf{t} \leftarrow \mathrm{MSB}(\mathbf{A}\mathbf{s}_1 + \mathbf{s}_2)$
3. $\mathrm{sk} := (\mathbf{s}_1, \mathbf{s}_2), \mathrm{vk} := \mathbf{t}$

Sign(msg, sk)

1. $\mathbf{r} \leftarrow \chi_3$ (short)
2. $\mathbf{u} \leftarrow \mathrm{MSB}(\mathbf{A}\mathbf{r})$
3. $\mathbf{c} \leftarrow H(\mathbf{u} \| \mathrm{msg})$ (short)
4. $\mathbf{z} \leftarrow \mathbf{r} - \mathbf{c}\mathbf{s}_1$
5. Rejection sampling step
6. $\mathrm{sig} := (\mathbf{u}, \mathbf{z})$

Verify(msg, vk, sig)

1. Accept iff $\mathbf{z}$ is short and $\mathrm{MSB}(\mathbf{A}\mathbf{z} - \mathbf{t}\mathbf{c}) = \mathbf{u}$

Dilithium trick [LDK+17] (naive version):
the signer drops the least significant bits of $\mathbf{t}$ during **Keygen**.

- → vk gets shorter.
- → Intuitively, this adds an error term $\mathbf{e}$ to $\mathbf{t}$
- → $\mathbf{A}\mathbf{z} - \mathbf{c}\mathbf{t} = \mathbf{u} - \underline{\mathbf{c}(\mathbf{s}_2 + \mathbf{e})}$

With mild bit dropping, the signature is valid with good probability (if it isn't, restart).

Dilithium uses a more sophisticated version of this trick.

A closer look at rejection sampling (here with SIS)

Sign(msg, sk)

1. Sample $\mathbf{r}$ uniformly in $\{-R, R\}^n$
2. $\mathbf{u} \leftarrow \mathbf{A}\mathbf{r}$
3. $\mathbf{c} \leftarrow H(\mathbf{u} \| \text{msg})$ (short)
4. $\mathbf{z} \leftarrow \mathbf{r} - \mathbf{c}\mathbf{s}$
5. Rejection sampling step
6. $\text{sig} := (\mathbf{u}, \mathbf{z})$

How do we choose the distribution of $\mathbf{r}$ and perform rejection sampling? Suppose:

- $\mathbf{r}$ is sampled uniformly in $\{-R, \ldots, R\}^n$
- $\mathbf{c}\mathbf{s}_1$ is guaranteed to be in $\{-S, \ldots, S\}^n$

A closer look at rejection sampling (here with SIS)

PQSHIELD

Sign(msg, sk)

1. Sample $\mathbf{r}$ uniformly in $\{-R, R\}^n$
2. $\mathbf{u} \leftarrow \mathbf{A}\mathbf{r}$
3. $\mathbf{c} \leftarrow H(\mathbf{u} \| \mathsf{msg})$ (short)
4. $\mathbf{z} \leftarrow \mathbf{r} - \mathbf{c}\mathbf{s}$
5. Rejection sampling step
6. $\mathsf{sig} := (\mathbf{u}, \mathbf{z})$

How do we choose the distribution of $\mathbf{r}$ and perform rejection sampling? Suppose:

- → $\mathbf{r}$ is sampled uniformly in $\{-R, \dots, R\}^n$
- → $\mathbf{c}\mathbf{s}_1$ is guaranteed to be in $\{-S, \dots, S\}^n$

Does a transcript $(\mathbf{u}, \mathbf{c}, \mathbf{z})$ leak information?

✗ $\mathbf{z} \notin \{-R, \dots, R\}^n \Rightarrow \mathbf{z}$ leaks the "direction" of $\mathbf{c}\mathbf{s}_1$

✓ $\mathbf{z} \in \{-(R-S), \dots, (R-S)\}^n \Rightarrow \mathbf{z}$ leaks nothing. Indeed, for any $\mathbf{z}^*$ in this set:

$$\mathbb{P}[\mathbf{r} - \mathbf{c}\mathbf{s}_1 = \mathbf{z}^*] = \mathbb{P}[\mathbf{r} = \underbrace{\mathbf{z}^* + \mathbf{c}\mathbf{s}_1}_{\in\{-R,\dots,R\}^n}] = \frac{1}{(2R+1)^n}$$

A closer look at rejection sampling (here with SIS)

Sign(msg, sk)

1. Sample $\mathbf{r}$ uniformly in $\{-R, R\}^n$
2. $\mathbf{u} \leftarrow \mathbf{Ar}$
3. $\mathbf{c} \leftarrow H(\mathbf{u} \| \mathsf{msg})$ (short)
4. $\mathbf{z} \leftarrow \mathbf{r} - \mathbf{cs}$
5. Rejection sampling step
6. $\mathsf{sig} := (\mathbf{u}, \mathbf{z})$

How do we choose the distribution of $\mathbf{r}$ and perform rejection sampling? Suppose:

- → $\mathbf{r}$ is sampled uniformly in $\{-R, \ldots, R\}^n$
- → $\mathbf{cs}_1$ is guaranteed to be in $\{-S, \ldots, S\}^n$

Does a transcript $(\mathbf{u}, \mathbf{c}, \mathbf{z})$ leak information?

- ✗ $\mathbf{z} \notin \{-R, \ldots, R\}^n \Rightarrow \mathbf{z}$ leaks the “direction” of $\mathbf{cs}_1$
- ✗ $\mathbf{z} \in \{-R, \ldots, R\}^n \backslash \{-(R-S), \ldots, (R-S)\}^n \Rightarrow$ more subtle but also leaks
- ✓ $\mathbf{z} \in \{-(R-S), \ldots, (R-S)\}^n \Rightarrow \mathbf{z}$ leaks nothing. Indeed, for any $\mathbf{z}^*$ in this set:

$$\mathbb{P}[\mathbf{r} - \mathbf{cs}_1 = \mathbf{z}^*] = \mathbb{P}[\mathbf{r} = \underbrace{\mathbf{z}^* + \mathbf{cs}_1}_{\in\{-R,\ldots,R\}^n}] = \frac{1}{(2R+1)^n}$$

A closer look at rejection sampling (here with SIS)

Sign(msg, sk)

1. Sample $\mathbf{r}$ uniformly in $\{-R, R\}^n$
2. $\mathbf{u} \leftarrow \mathbf{A}\mathbf{r}$
3. $\mathbf{c} \leftarrow H(\mathbf{u} \| \mathrm{msg})$ (short)
4. $\mathbf{z} \leftarrow \mathbf{r} - \mathbf{c}\mathbf{s}$
5. If $\|\mathbf{z}\|_\infty > R - S$, goto 1
6. $\mathrm{sig} := (\mathbf{u}, \mathbf{z})$

How do we choose the distribution of $\mathbf{r}$ and perform rejection sampling? Suppose:

- → $\mathbf{r}$ is sampled uniformly in $\{-R, \ldots, R\}^n$
- → $\mathbf{c}\mathbf{s}_1$ is guaranteed to be in $\{-S, \ldots, S\}^n$

Does a transcript $(\mathbf{u}, \mathbf{c}, \mathbf{z})$ leak information?

- ✗ $\mathbf{z} \notin \{-R, \ldots, R\}^n \Rightarrow \mathbf{z}$ leaks the "direction" of $\mathbf{c}\mathbf{s}_1$
- ✗ $\mathbf{z} \in \{-R, \ldots, R\}^n \backslash \{-(R-S), \ldots, (R-S)\}^n \Rightarrow$ more subtle but also leaks
- ✓ $\mathbf{z} \in \{-(R-S), \ldots, (R-S)\}^n \Rightarrow \mathbf{z}$ leaks nothing. Indeed, for any $\mathbf{z}^*$ in this set:

$$\mathbb{P}[\mathbf{r} - \mathbf{c}\mathbf{s}_1 = \mathbf{z}^*] = \mathbb{P}[\mathbf{r} = \underbrace{\mathbf{z}^* + \mathbf{c}\mathbf{s}_1}_{\in\{-R,\ldots,R\}^n}] = \frac{1}{(2R+1)^n}$$

Accept if $\mathbf{z} \in \{-(R-S), \ldots, (R-S)\}^n$. This happens w/ prob. $\approx \left(1 - \frac{S}{R}\right)^n \leq \exp\left(-\frac{S}{nR}\right)$.

Section conclusion and bibliography

Foundations (FSwA)

- Using SIS [Lyu09]
- Using SIS + LWE [Lyu12]

Ninja tricks

- Cutting |**sig**| [BG14]
- Cutting |**vk**| [LDK+17]

Distributions

- In-depth survey [DFPS22]
- Bimodal Gaussians [DDLL13]

Efficient instantiations

- Dilithium [LDK+17]
- qTESLA [BAA+17]
- BLISS [DDLL13]

Thank You!

Nina Bindel, Sedat Akleylek, Erdem Alkim, Paulo S. L. M. Barreto, Johannes Buchmann, Edward Eaton, Gus Gutoski, Juliane Kramer, Patrick Longa, Harun Polat, Jefferson E. Ricardini, and Gustavo Zanon.
qTESLA.
Technical report, National Institute of Standards and Technology, 2017.
available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.

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, Heidelberg, February 2014.

Yilei Chen, Nicholas Genise, and Pratyay Mukherjee.
Approximate trapdoors for lattices and smaller hash-and-sign signatures.
In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019*, *Part III*, volume 11923 of *LNCS*, pages 3–32. Springer, Heidelberg, December 2019.

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, Heidelberg, August 2013.

Julien Devevey, Omar Fawzi, Alain Passelègue, and Damien Stehlé.
On rejection sampling in lyubashevsky's signature scheme.

Cryptology ePrint Archive, Paper 2022/1249, 2022.
https://eprint.iacr.org/2022/1249.

Léo Ducas, Vadim Lyubashevsky, and Thomas Prest.
Efficient identity-based encryption over NTRU lattices.
In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part II*, volume 8874 of *LNCS*, pages 22–41. Springer, Heidelberg, December 2014.

Léo Ducas and Thomas Prest.
Fast fourier orthogonalization.
In Sergei A. Abramov, Eugene V. Zima, and Xiao-Shan Gao, editors, *Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation, ISSAC 2016, Waterloo, ON, Canada, July 19-22, 2016*, pages 191–198. ACM, 2016.

Thomas Espitau et al.
On hash-and-sign lattice-based signatures: Gpv, falcon and beyond.
ENS Crypto Seminar, 2022.
https://crypto.di.ens.fr/seminars:main.

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 falcon.
In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part III*, volume 13277 of *LNCS*, pages 222–253. Springer, Heidelberg, May / June 2022.

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.

Jeffrey Hoffstein, Nick Howgrave-Graham, Jill Pipher, Joseph H. Silverman, and William Whyte.
NTRUSIGN: Digital signatures using the NTRU lattice.
In Marc Joye, editor, *CT-RSA 2003*, volume 2612 of *LNCS*, pages 122–140. Springer, Heidelberg, April 2003.

Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Peter Schwabe, Gregor Seiler, and Damien Stehlé.
CRYSTALS-DILITHIUM.
Technical report, National Institute of Standards and Technology, 2017.
available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.

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, Heidelberg, December 2009.

Vadim Lyubashevsky.
Lattice signatures without trapdoors.
In Pointcheval and Johansson [PJ12], pages 738–755.

Daniele Micciancio and Chris Peikert.
Trapdoors for lattices: Simpler, tighter, faster, smaller.
In Pointcheval and Johansson [PJ12], pages 700–718.

Chris Peikert.
An efficient and parallel Gaussian sampler for lattices.
In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 80–97. Springer, Heidelberg, August 2010.

Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang.
FALCON.
Technical report, National Institute of Standards and Technology, 2017.
available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`.

David Pointcheval and Thomas Johansson, editors.
EUROCRYPT 2012, volume 7237 of *LNCS*. Springer, Heidelberg, April 2012.

Thomas Prest.
Gaussian Sampling in Lattice-Based Cryptography.
PhD thesis, École Normale Supérieure, Paris, France, 2015.

Thomas Prest.
Sharper bounds in lattice-based cryptography using the Rényi divergence.

In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017*, *Part I*, volume 10624 of *LNCS*, pages 347–374. Springer, Heidelberg, December 2017.