Lattice-based NIST Candidates - Abstractions and Ninja Tricks

Thomas Prest

PQShield

European Cyber Week

What is PQShield?

PQShield is a start-up specialised in post-quantum cryptography (PQC).
We have a legal presence in UK/NL/FR/US, and offices in UK/FR.

A few products/projects:

PQSOC

- Hardware for low/high-end devices
- PQC math & symmetric copropressors
- Modular IP or self-contained solution
- Efficient SCA mitigations
- FIPS 140-3 ready

PQSlib

- Lightweight PQC library
- Firmware for embedded & IoT

PQSDK

- Crypto SDK for mobile & servers
- For application such as PKI, TLS, VPN
- FIPS 140-3 ready

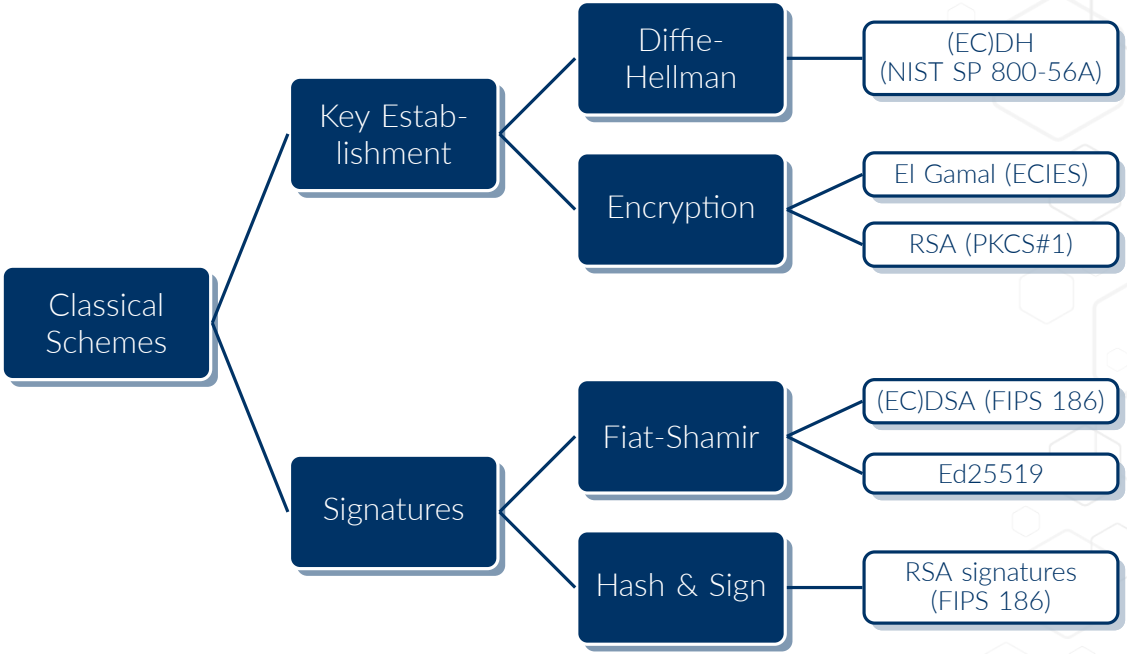
PQE2E

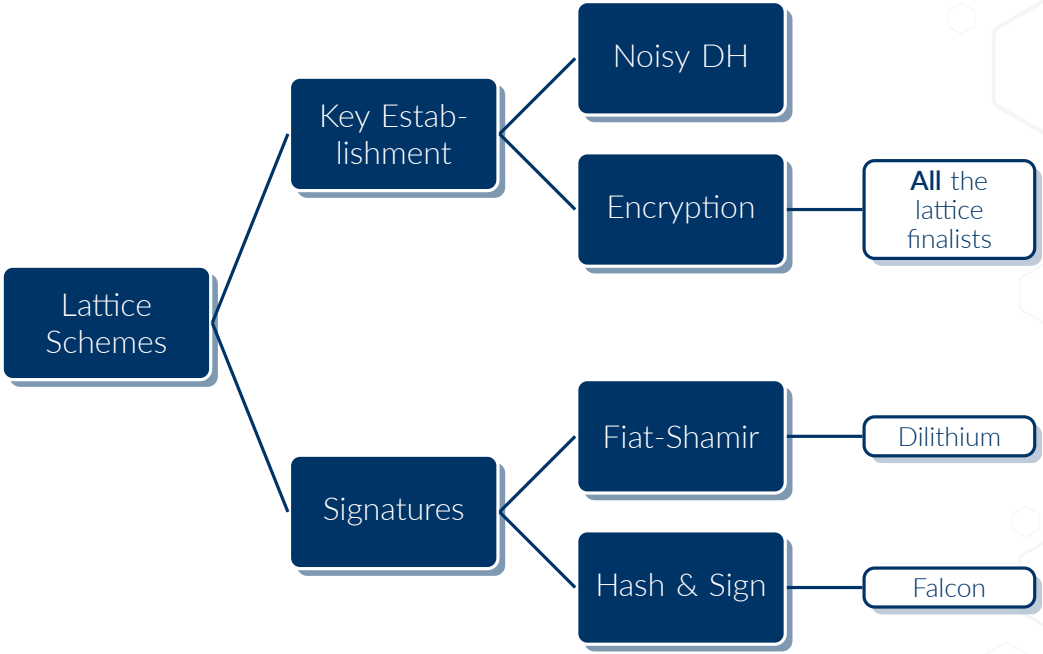
- SDK for end-to-end encryption
- PQ alternative to WhatsApp, Citadel
- Leverages our provably secure, peer-reviewed, PQC protocol

Roadmap

- We proceed by **analogy** between classical vs lattice-based schemes (e.g. El Gamal vs “noisy El Gamal” [LPR10, LP11])
- This **abstraction** allow to re-use knowledge and intuition of existing schemes
- We can still take advantage of the specificites of lattice via some **ninja tricks**

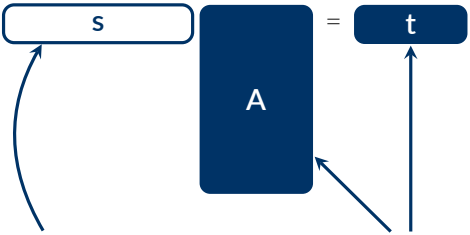
Introduction


Classical Schemes
Key Estab-lishment
Diffie-Hellman
(EC)DH (NIST SP 800-56A)
Encryption
El Gamal (ECIES)
RSA (PKCS#1)
Signatures
Fiat-Shamir
(EC)DSA (FIPS 186)
Ed25519
Hash & Sign
RSA signatures (FIPS 186)

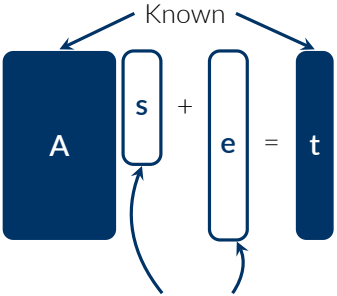

Lattice Schemes
Key Estab-lishment
Noisy DH
Encryption
All the lattice finalists
Signatures
Fiat-Shamir
Dilithium
Hash & Sign
Falcon

Selected problems

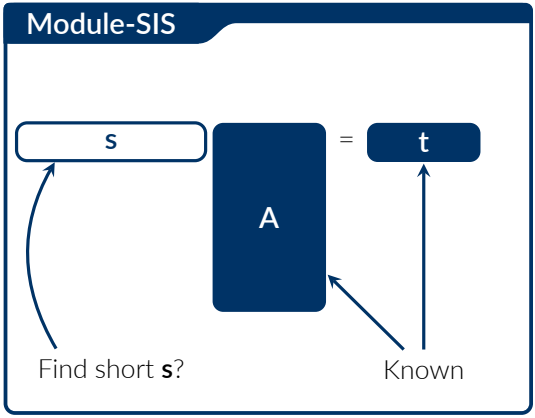
Module-SIS

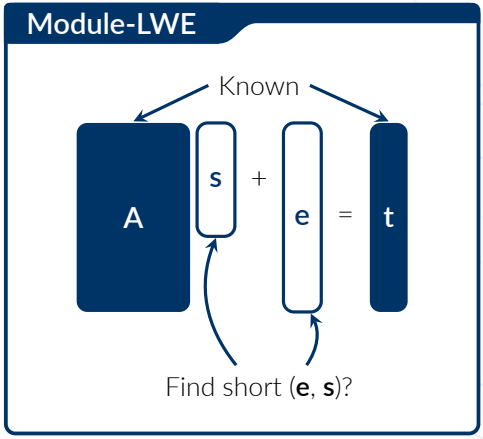


Module-LWE



Selected problems



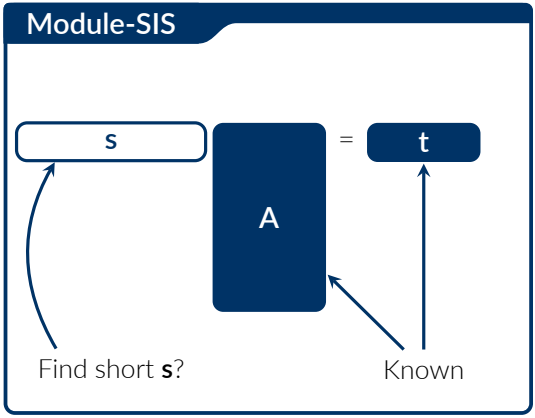


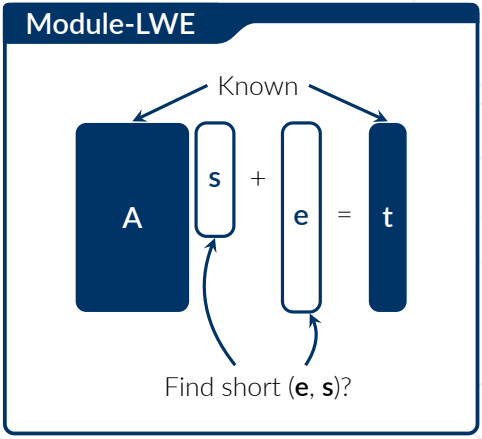
Parameters we can play on:

- **Base ring $\mathcal{R}_q$:** modular ring $\mathbb{Z}_q$? Polynomial ring $\mathbb{Z}_q[x]/(x^d + 1)$?
- **Dimensions** of the matrices and vectors
- **What does "short" mean?** With respect to the Euclidean norm? Infinity norm?

Also variants with more fundamental changes, for example LWR.

Selected problems





We will constantly draw analogies between DLOG and SIS/LWE:

- **DLOG:** Given (g, g^a), find a.
- **SIS:** Given $(\mathbf{A}, \mathbf{s}^t\mathbf{A})$, find $\mathbf{s}$.
- **LWE:** Given $(\mathbf{A}, \mathbf{As} + \mathbf{e})$, find $(\mathbf{s}, \mathbf{e})$.

Noisy El Gamal

El Gamal

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

1. Sample $x \leftarrow \mathbb{Z}_{|\mathbb{G}|}$
2. $h \leftarrow g^x$
3. $\mathsf{sk} := x, \mathsf{pk} := h$

Enc$(\mathtt{msg}, \mathsf{pk})$

1. Sample $r \leftarrow \mathbb{Z}_{|\mathbb{G}|}$
2. $u \leftarrow g^r$
3. $v \leftarrow h^r \cdot \mathtt{msg}$
4. $\mathsf{c} := (u, v)$

Dec$(\mathsf{c}, \mathsf{sk})$

1. $\mathtt{msg} \leftarrow v \cdot u^{-x}$

"Noisy" El Gamal [LPR10, LP11]

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

1. Sample short $\mathbf{S}, \mathbf{E}$
2. $\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$
3. $\mathsf{sk} := (\mathbf{S}, \mathbf{E}), \mathsf{pk} := \mathbf{B}$

Enc$(\mathtt{msg}, \mathsf{pk})$

1. Sample short $\mathbf{R}, \mathbf{E}', \mathbf{E}''$
2. $\mathbf{U} \leftarrow \mathbf{RA} + \mathbf{E}'$
3. $\mathbf{V} \leftarrow \mathbf{RB} + \mathbf{E}'' + \mathtt{Encode}(\mathtt{msg})$
4. $\mathsf{c} := (\mathbf{U}, \mathbf{V})$

Dec$(\mathsf{c}, \mathsf{sk})$

1. $\mathtt{msg} \leftarrow \mathtt{Decode}(\mathbf{V} - \mathbf{US})$

Decryption is successful since:

- **El Gamal:** $v \cdot u^{-x} = (h^r \cdot \mathtt{msg}) \cdot (g^r)^{-x} = \mathtt{msg}$
- **Noisy El Gamal:**

$$\begin{aligned} \mathbf{V} - \mathbf{US} &= \big(\mathbf{R}(\mathbf{AS} + \mathbf{E}) + \mathbf{E}'' + \mathtt{Encode}(\mathtt{msg})\big) - (\mathbf{RA} + \mathbf{E}')\mathbf{S} && (1) \\ &= \mathtt{Encode}(\mathtt{msg}) + (\mathbf{RE} + \mathbf{E}'' - \mathbf{E}'\mathbf{S}) && (2) \end{aligned}$$

The recipient recovers $\mathtt{msg}$ as long as $(\mathbf{RE} + \mathbf{E}'' - \mathbf{E}'\mathbf{S})$ remains small.

Obtaining security against active attackers

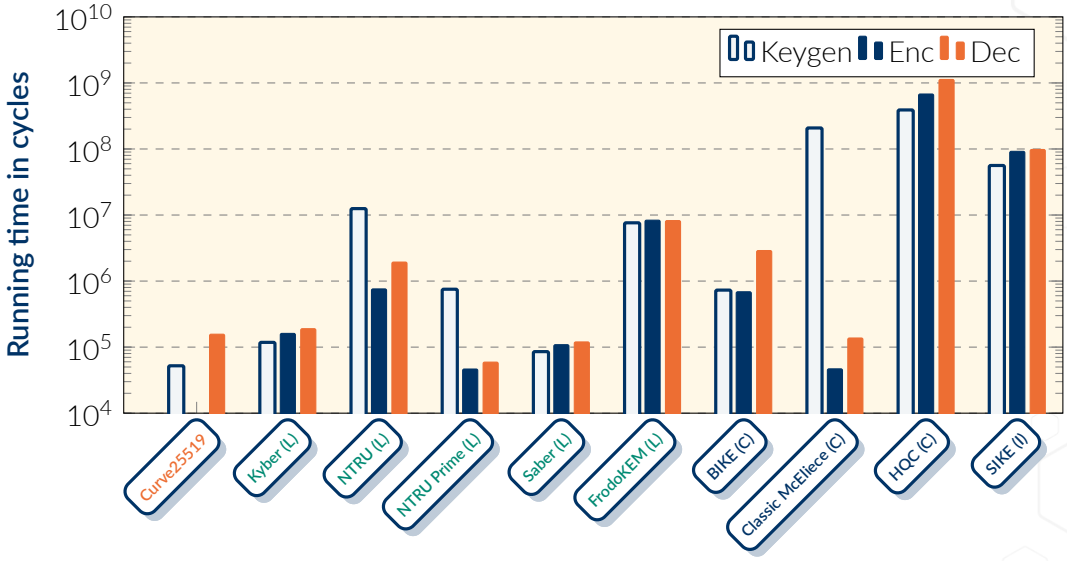
Most schemes rely on transforms in [HHK17]:

- → Variants of Fujisaki-Okamoto, handle decryption failures [DRV20]
- → Tight proofs in the ROM but not the QROM

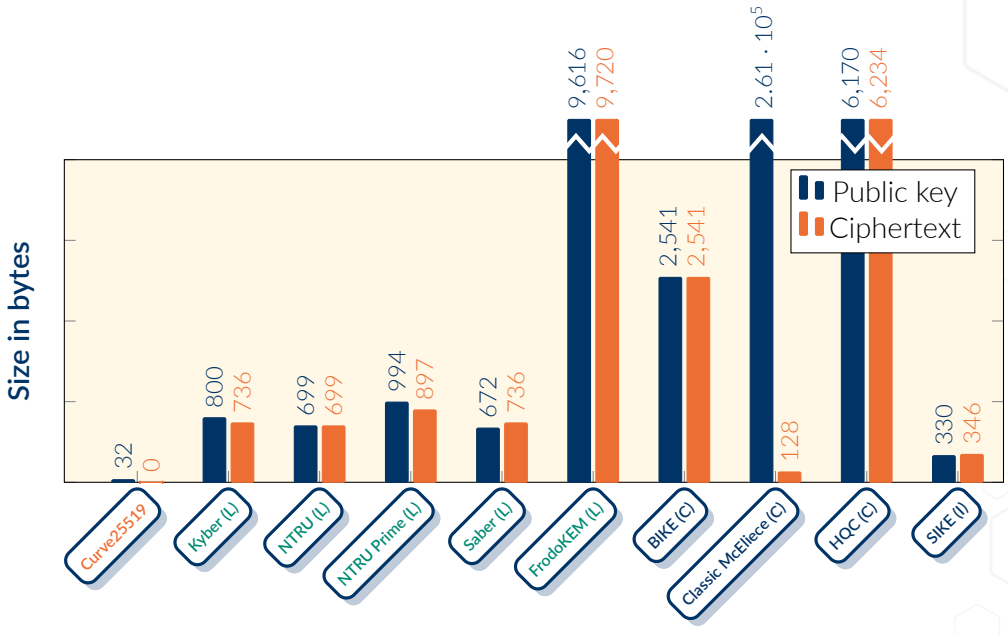
NTRU and NTRU Prime use [BP18] and [Den03] instead:

- → Do not require re-encryption
- → Tight proofs in the QROM (under non-standard assumptions)

Computation Cost of Level 1 KEMs



Bandwidth cost of NIST Level 1 KEMs



Multi-Recipient Encryption

Figure 1: Broadcast

Figure 2: Group Messaging

- Post-quantum cryptography can be expensive.
- This can be amplified when large groups of users are involved.
- When encrypting a message to *N* recipients, can we do better than sending *N* ciphertexts?

In LWE/LWR proposals, $\mathbf{U}$ does almost not depend on the public key.

- → Use the same $\mathbf{A}$ for all public keys.
- → Use the same $\mathbf{U}$ when encrypting **the same `msg`** to several recipients.

Enc$(\mathtt{msg}, \mathbf{pk} = (\mathbf{A}, \mathbf{B}))$

1. $\mathbf{R}, \mathbf{E}', \mathbf{E}'' \leftarrow \chi_3 \times \chi_4 \times \chi_5$
2. $\mathbf{U} \leftarrow \mathbf{RA} + \mathbf{E}'$
3. $\mathbf{V} \leftarrow \mathbf{RB} + \mathbf{E}'' + \mathtt{Encode}(\mathtt{msg})$
4. $c := (\mathbf{U}, \mathbf{V})$

In LWE/LWR proposals, $\mathbf{U}$ does almost not depend on the public key.

- → Use the same $\mathbf{A}$ for all public keys.
- → Use the same $\mathbf{U}$ when encrypting **the same `msg`** to several recipients.

`Enc`$(\mathtt{msg}, \mathbf{pk} = (\mathbf{A}, \mathbf{B}))$

1. $\mathbf{R}, \mathbf{E}', \mathbf{E}'' \leftarrow \chi_3 \times \chi_4 \times \chi_5$
2. $\mathbf{U} \leftarrow \mathbf{RA} + \mathbf{E}'$
3. $\mathbf{V} \leftarrow \mathbf{RB} + \mathbf{E}'' + \mathtt{Encode}(\mathtt{msg})$
4. $\mathtt{c} := (\mathbf{U}, \mathbf{V})$

$\Longrightarrow$

`MultiEnc`$(\mathtt{msg}, \mathbf{pk}_1, \ldots, \mathbf{pk}_k)$

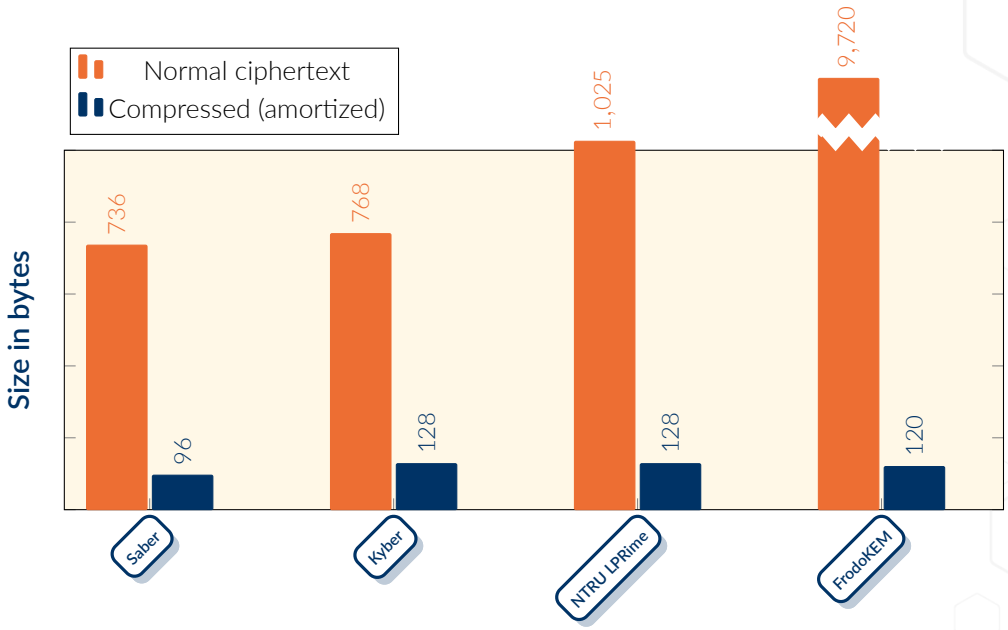
1. $\mathbf{R}, \mathbf{E}' \leftarrow \chi_3 \times \chi_4$
2. $\mathbf{U} \leftarrow \mathbf{RA} + \mathbf{E}'$
3. For $i = 1, \ldots, k$:
   1. $\mathbf{E}_i'' \leftarrow \chi_5$
   2. $\mathbf{V}_i \leftarrow \mathbf{RB}_i + \mathbf{E}_i'' + \mathtt{Encode}(\mathtt{msg})$
4. $\mathtt{c} := (\mathbf{U}, \mathbf{V}_1, \ldots, \mathbf{V}_k)$

This improves amortized costs by **factors up to 169**.

- → Faster encryption
- → Smaller ciphertexts

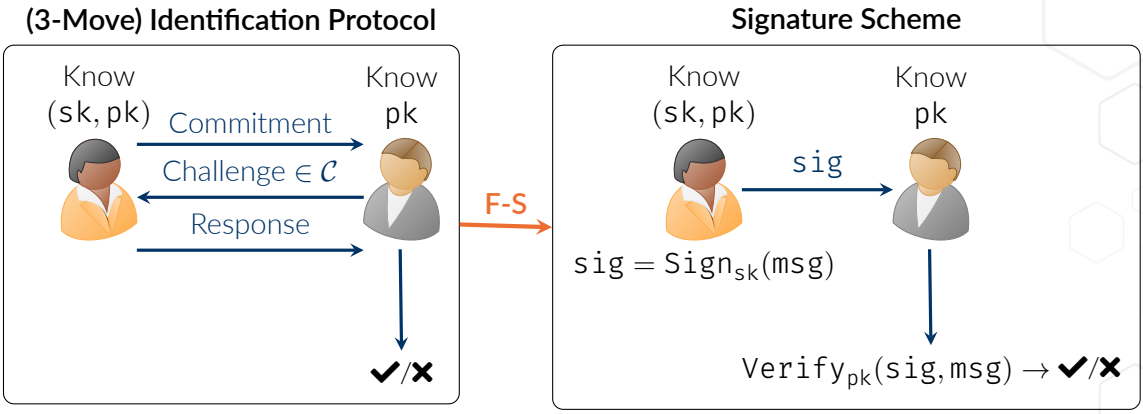
For more details, see our article [KKPP20].

Impact on Potential NIST Standards (Level I)



Fiat-Shamir Signatures

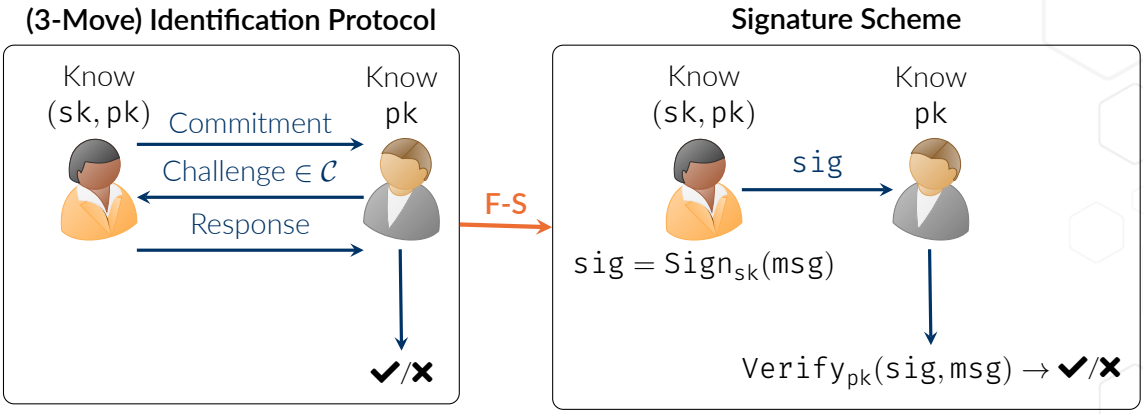
Fiat-Shamir Signatures



F-S refers to the Fiat-Shamir transform:

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

Fiat-Shamir Signatures



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 $\mathtt{sk}$
2. **Honest verifier zero-knowledge:** A valid transcript leaks no information about $\mathtt{sk}$
3. **Soundness:** A dishonest prover cannot convince a verifier he knows $\mathtt{sk}$

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. $\mathtt{sk} := x, \mathtt{pk} := h$

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

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

Verify$(\mathtt{msg}, \mathtt{pk})$

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. $\mathbf{s} \leftarrow \chi$ (short)
2. $\mathbf{t} \leftarrow \mathbf{A}\mathbf{s}$
3. $\mathtt{sk} := \mathbf{s}, \mathtt{pk} := \mathbf{t}$

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

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

Verify$(\mathtt{msg}, \mathtt{pk}, \mathtt{sig})$

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. $\mathbf{s} \leftarrow \chi$ (short)
2. $\mathbf{t} \leftarrow \mathbf{A}\mathbf{s}$
3. $\mathtt{sk} := \mathbf{s}, \mathtt{pk} := \mathbf{t}$

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

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

Verify$(\mathtt{msg}, \mathtt{pk}, \mathtt{sig})$

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. $\mathbf{s} \leftarrow \chi$ (short)
2. $\mathbf{t} \leftarrow \mathbf{A}\mathbf{s}$
3. $\mathtt{sk} := \mathbf{s}, \mathtt{pk} := \mathbf{t}$

Sign$(\mathtt{msg}, \mathbf{sk})$

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

Verify$(\mathtt{msg}, \mathbf{pk}, \mathtt{sig})$

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

Soundness: Using rewinding:

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

$$[\, \mathbf{A} \,\|\, \mathbf{t} \,] \cdot \begin{bmatrix} \mathbf{z} - \mathbf{z}' \\ \mathbf{c} - \mathbf{c}' \end{bmatrix} = \mathbf{0} \tag{3}$$

- ✓ **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. $\mathbf{s} \leftarrow \chi$ (short)
2. $\mathbf{t} \leftarrow \mathbf{As}$
3. $\mathtt{sk} := \mathbf{s}, \mathtt{pk} := \mathbf{t}$

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

1. $\mathbf{r} \leftarrow \chi$ (short)
2. $\mathbf{u} \leftarrow \mathbf{Ar}$
3. $\mathbf{c} \leftarrow H(\mathbf{u} \| \mathtt{msg})$ (short)
4. $\mathbf{z} \leftarrow \mathbf{r} - \mathbf{cs}$
5. Rejection sampling step
6. $\mathtt{sig} := (\mathbf{u}, \mathbf{z})$

Verify$(\mathtt{msg}, \mathtt{pk}, \mathtt{sig})$

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

- ✔ **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. $\mathbf{s} \leftarrow \chi$ (short)
2. $\mathbf{t} \leftarrow \mathbf{As}$
3. $\mathtt{sk} := \mathbf{s}, \mathtt{pk} := \mathbf{t}$

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

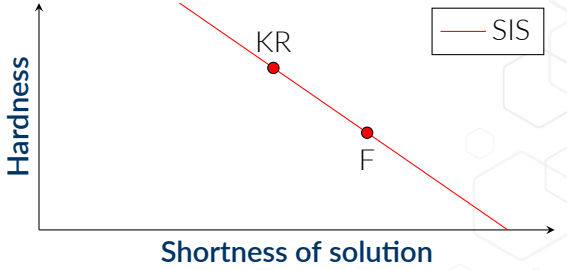
1. $\mathbf{r} \leftarrow \chi$ (short)
2. $\mathbf{u} \leftarrow \mathbf{Ar}$
3. $\mathbf{c} \leftarrow H(\mathbf{u} \| \mathtt{msg})$ (short)
4. $\mathbf{z} \leftarrow \mathbf{r} - \mathbf{cs}$
5. Rejection sampling step
6. $\mathtt{sig} := (\mathbf{u}, \mathbf{z})$

Verify$(\mathtt{msg}, \mathtt{pk}, \mathtt{sig})$

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

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. $\mathsf{sk} := (\mathbf{s}_1, \mathbf{s}_2), \mathsf{pk} := \mathbf{t}$

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

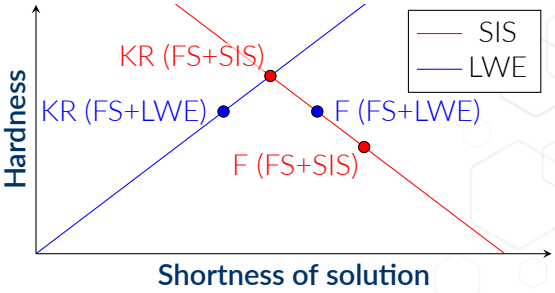
1. $\mathbf{r}_1, \mathbf{r}_2 \leftarrow \chi_3 \times \chi_4$ (short)
2. $\mathbf{u} \leftarrow \mathbf{A}\mathbf{r}_1 + \mathbf{r}_2$
3. $\mathbf{c} \leftarrow H(\mathbf{u} \| \mathsf{msg})$ (short)
4. $\mathbf{z}_1 \leftarrow \mathbf{r}_1 - \mathbf{c}\mathbf{s}_1$
5. $\mathbf{z}_2 \leftarrow \mathbf{r}_2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

Verify$(\mathbf{msg}, \mathbf{pk}, \mathbf{sig})$

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. $\mathtt{sk} := (\mathbf{s}_1, \mathbf{s}_2), \mathtt{pk} := \mathbf{t}$

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

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. $\mathbf{c} \leftarrow H(\mathbf{u} \| \mathtt{msg})$ (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. $\mathtt{sig} := (\mathbf{u}, \mathbf{z}_1, \mathbf{z}_2)$

Verify$(\mathtt{msg}, \mathtt{pk}, \mathtt{sig})$

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

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. $\mathsf{sk} := (\mathbf{s}_1, \mathbf{s}_2), \mathsf{pk} := \mathbf{t}$

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

1. $\mathbf{r} \leftarrow \chi_3$ (short)
2. $\mathbf{u} \leftarrow \mathrm{MSB}(\mathbf{Ar})$
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$(\mathbf{msg}, \mathbf{pk}, \mathbf{sig})$

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

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

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. $\mathtt{sk} := (\mathbf{s}_1, \mathbf{s}_2), \mathtt{pk} := \mathbf{t}$

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

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

Verify$(\mathtt{msg}, \mathtt{pk}, \mathtt{sig})$

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`.

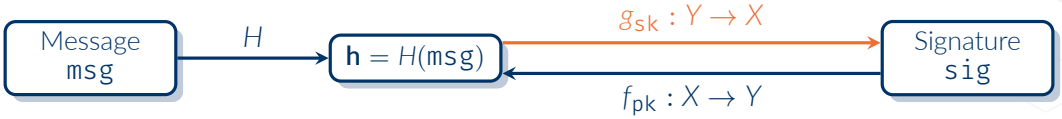
- → `pk` gets shorter.
- → Intuitively, this adds an error term $\mathbf{e}$ to $\mathbf{t}$
- → $\mathbf{A}\mathbf{z} - \mathbf{c}\mathbf{t} = \mathbf{u} - \mathbf{c}(\mathbf{s}_2 + \mathbf{e})$

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.

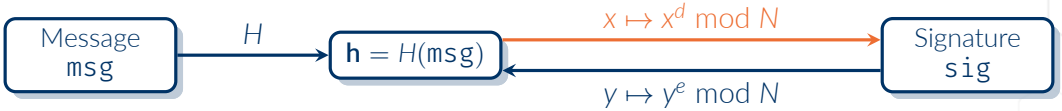
Hash-then-Sign

Signatures based on *Hash-then-Sign* (Simplified)



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

The case of RSA signatures



Example with **RSA signatures**:

- $g_{\mathsf{sk}}(x) = x^d \bmod N$, and $f_{\mathsf{pk}}(y) = y^e \bmod N$.
- $(f_{\mathsf{pk}}, g_{\mathsf{sk}})$ are often abstracted as trapdoor permutations [BR96, Cor02]:
  1. Given only pk, f_{pk} is hard to invert for (almost) all inputs.
  2. $f_{\mathsf{pk}} \circ g_{\mathsf{sk}} = Id$ and $X = Y$ (hence f_{pk} and g_{sk} are permutations).

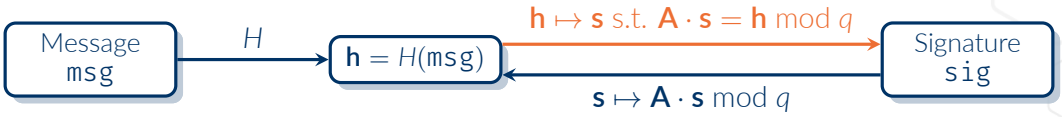
  We do not know how to realise this abstraction under PQ assumptions.

The case of lattices (first attempt)

Following [GGH97, HHP+03], let us try to instantiate this blueprint with lattices:

- *Verification key:* pk 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 $\mathbf{A} \cdot \mathbf{B} = \mathbf{0} \bmod q$.

The case of lattices (first attempt)



Recall ($\mathbf{A} \cdot \mathbf{B} = \mathbf{0} \bmod q$) and $\mathbf{B}$ is short.

→ *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 $\mathbf{v} := \mathbf{B} \cdot \lfloor \mathbf{B}^{-1} \cdot \mathbf{c} \rceil$
4. The signature is $\mathbf{s} := \mathbf{c} - \mathbf{v}$

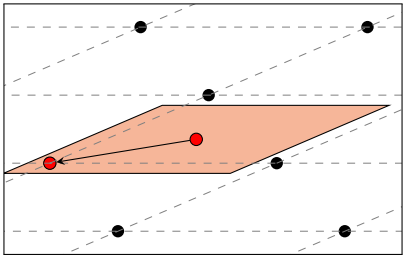
→ *Verification:*

1. Check that $\mathbf{A} \cdot \mathbf{s} = \mathbf{h}$.
2. Check that $\mathbf{s}$ is short

One can show that a honestly generated signature passes verification.

1. $\mathbf{A} \cdot \mathbf{s} = \mathbf{A} \cdot (\mathbf{c} - \mathbf{v}) = \overbrace{\mathbf{A} \cdot \mathbf{c}}^{=\mathbf{h}} - \overbrace{\mathbf{A} \cdot \mathbf{B}}^{=\mathbf{0}} \cdot \lfloor \mathbf{B}^{-1} \cdot \mathbf{c} \rceil = \mathbf{h}$
2. If $\mathbf{c} = \mathbf{B} \cdot \mathbf{t}$, then $\mathbf{s} = \mathbf{B} \cdot (\mathbf{t} - \lfloor \mathbf{t} \rceil) \in \mathbf{B} \cdot \left[-\frac{1}{2}, \frac{1}{2}\right]^m$. Since $\mathbf{B}$ is short, $\mathbf{s}$ is short.

What about security?

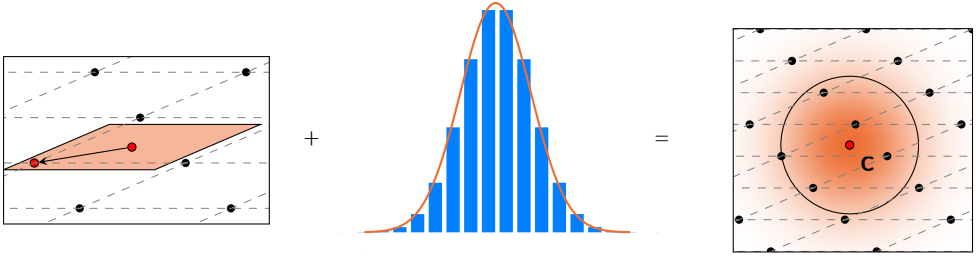


We have seen in the previous slide that $\mathtt{sig} \in \mathbf{B} \cdot \left[-\frac{1}{2}, \frac{1}{2}\right]^m$.

- This means signatures will leak the shape of the signing key $\mathtt{sk} = \mathbf{B}$.
- Exploited in statistical key-recovery attacks [NR06, DN12]

Solution: trapdoor sampling [GPV08]

By carefully randomising the signing procedure, we can make the signature distribution independent of the secret basis **B**.



Putting everything together (Falcon)

Keygen(1^λ)

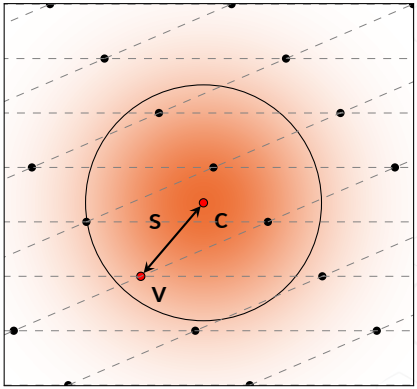
1. Gen. matrices $\mathbf{A}, \mathbf{B}$ s.t.:
   - $\mathbf{A} \cdot \mathbf{B} = 0$
   - $\mathbf{B}$ has small coefficients
2. $\mathtt{pk} := \mathbf{A}$, $\mathtt{sk} := \mathbf{B}$

Sign$(\mathtt{msg}, \mathtt{sk} = \mathbf{B})$

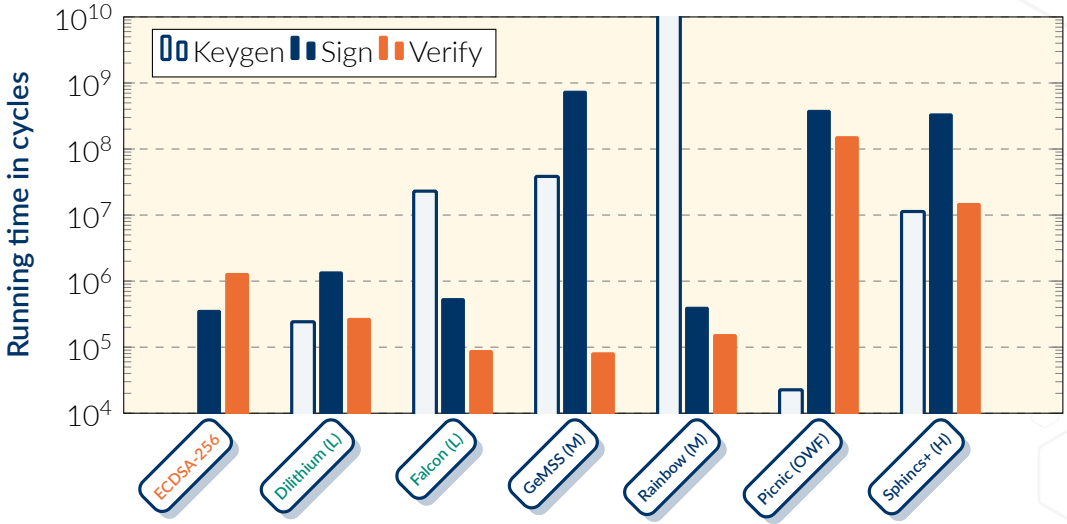
1. Compute $\mathbf{c}$ such that $\mathbf{A} \cdot \mathbf{c} = H(\mathtt{msg})$
2. $\mathbf{v} \leftarrow$ vector in $\Lambda(\mathbf{B})$, close to $\mathbf{c}$
3. $\mathtt{sig} := \mathbf{s} = (\mathbf{c} - \mathbf{v})$

Verify$(\mathtt{msg}, \mathtt{pk} = \mathbf{A}, \mathtt{sig} = \mathbf{s})$

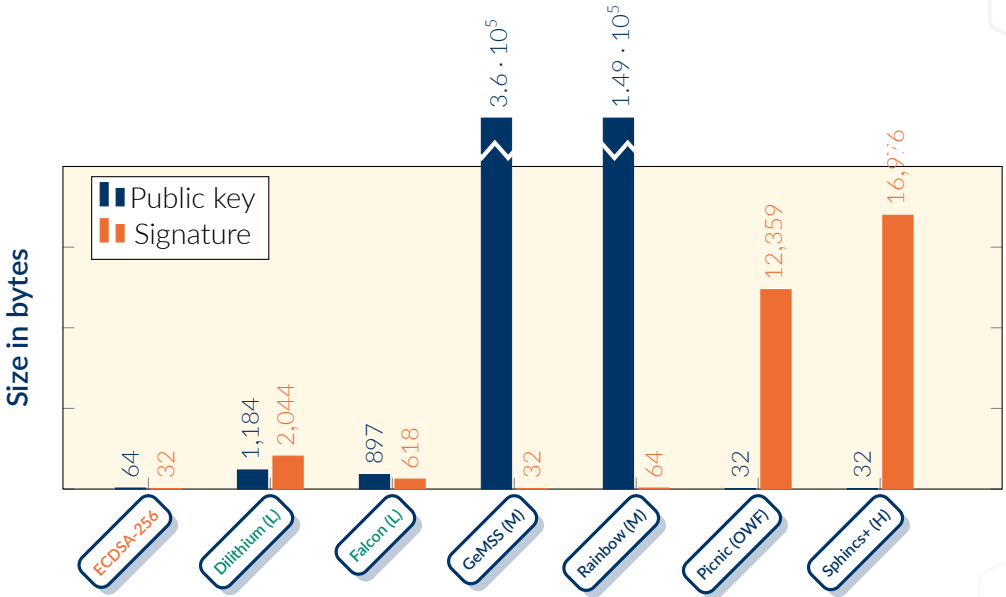
Check ($\mathbf{s}$ short) & ($\mathbf{A} \cdot \mathbf{s} = H(\mathtt{msg})$)



Computation Cost of NIST Level 1 Signatures



Bandwidth cost of NIST Level 1 Signatures



Conclusion

Classical and lattice-based schemes share many similarities:

- *Same paradigms:* Schnorr-type identification protocols, El Gamal-type encryption, full-domain hash, etc.
- The analogy sometimes fails, but this is also informative

For a more comprehensive overview of lattice-based cryptography, see Simon's Institute 2020 programme "*Lattices: Algorithms, Complexity, and Cryptography*": https://simons.berkeley.edu/programs/lattices2020.

Thank You

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.

Daniel J. Bernstein and Edoardo Persichetti.
Towards KEM unification.
Cryptology ePrint Archive, Report 2018/526, 2018.
https://eprint.iacr.org/2018/526.

Mihir Bellare and Phillip Rogaway.
The exact security of digital signatures: How to sign with RSA and Rabin.
In Ueli M. Maurer, editor, EUROCRYPT'96, volume 1070 of LNCS, pages 399–416. Springer, Heidelberg, May 1996.

Jean-Sébastien Coron.
Optimal security proofs for PSS and other signature schemes.
In Lars R. Knudsen, editor, EUROCRYPT 2002, volume 2332 of LNCS, pages 272–287. Springer, Heidelberg, April / May 2002.

Alexander W. Dent.
A designer's guide to KEMs.
In Kenneth G. Paterson, editor, 9th IMA International Conference on Cryptography and Coding, volume 2898 of LNCS, pages 133–151. Springer, Heidelberg, December 2003.

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, Heidelberg, December 2012.

Jan-Pieter D'Anvers, Mélissa Rossi, and Fernando Virdia.
(One) failure is not an option: Bootstrapping the search for failures in lattice-based encryption schemes.
In Anne Canteaut and Yuval Ishai, editors, EUROCRYPT 2020, Part III, volume 12107 of LNCS, pages 3–33. Springer, Heidelberg, May 2020.

Oded Goldreich, Shafi Goldwasser, and Shai Halevi.
Public-key cryptosystems from lattice reduction problems.
In Burton S. Kaliski Jr., editor, CRYPTO'97, volume 1294 of LNCS, pages 112–131. Springer, Heidelberg, August 1997.

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.

Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz.
A modular analysis of the Fujisaki-Okamoto transformation.
In Yael Kalai and Leonid Reyzin, editors, TCC 2017, Part I, volume 10677 of LNCS, pages 341–371. Springer, Heidelberg, November 2017.

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.

Shuichi Katsumata, Kris Kwiatkowski, Federico Pintore, and Thomas Prest.
Scalable ciphertext compression techniques for post-quantum KEMs and their applications.
In Shiho Moriai and Huaxiong Wang, editors, ASIACRYPT 2020, Part I, volume 12491 of LNCS, pages 289–320. Springer, Heidelberg, December 2020.

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.

Richard Lindner and Chris Peikert.
Better key sizes (and attacks) for LWE-based encryption.
In Aggelos Kiayias, editor, CT-RSA 2011, volume 6558 of LNCS, pages 319–339. Springer, Heidelberg, February 2011.

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, Heidelberg, May / June 2010.

Vadim Lyubashevsky.

Lattice signatures without trapdoors.

In David Pointcheval and Thomas Johansson, editors, EUROCRYPT 2012, volume 7237 of LNCS, pages 738–755. Springer, Heidelberg, April 2012.

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, Heidelberg, May / June 2006.