# On Structure-Preserving Cryptography and Lattices

Dennis Hofheinz[1], Kristina Hostáková[1], Roman Langrehr[1], Bogdan Ursu[2]

[1] ETH Zurich [2]Linea, Consensys

# Structure-Preserving Cryptography (SPS)

The Groth-Sahai NIZK [GS08] allows to efficiently prove quadratic relations over:

- the group elements of a bilinear pairing group.
- and its exponents

# Structure-Preserving Cryptography (SPS)

The Groth-Sahai NIZK [GS08] allows to efficiently prove quadratic relations over:

- the group elements of a bilinear pairing group.
- its exponents

The structure of signatures and encryption in pairing groups is compatible with these quadratic relations.

structure-preserving cryptography.

# Structure-Preserving Cryptography (SPS)

In structure-preserving cryptography, the NIZK efficiently proves statements of the form:

- a ciphertext encrypts a valid signature. ← focus on this, the other case will become clear along the way.

- a signature signs a valid ciphertext.

# Structure-Preserving Cryptography (SPS)

In structure-preserving cryptography, the NIZK efficiently proves statements of the form:

- a ciphertext encrypts a valid signature. ← focus on this, the other case will become clear along the way.
- a signature signs a valid ciphertext.

[GS08] NIZK is in the <u>standard model</u>
perfect correctness

→ Encryption and signing can be nested indefinitely.

# Structure-Preserving Cryptography (SPS)

In structure-preserving cryptography, the NIZK efficiently proves statements of the form:

- a ciphertext encrypts a valid signature. ← focus on this, the other case will become clear along the way.

- a signature signs a valid ciphertext.

[GS08] NIZK is in the standard model
perfect correctness

→ Encryption and signing can be nested indefinitely.

All operations lie in the same group, allowing for native arithmetic.

avoid generic NIZKs and expensive circuitry.

Linea

# Motivation

Structure-preserving primitives enable or enhance a wide range of constructions:

- Verifiably Encrypted Signatures
- Delegatable Anonymous Credentials
- Group Signatures
- Ring Signatures

# Motivation

Can we get similar structure preserving cryptography for lattices?

# Motivation

Can we get similar structure preserving cryptography for lattices?

We formalise unifying notions shared by a family of encryption and signatures schemes

Provide a NIZK that is compatible with this notion

standard model security

Linea

# Example: Regev Encryption [Regev05]

KeyGen

Sample uniform matrix $\mathbf{A}$, uniform $\mathbf{s}$ and $\mathbf{e} \leftarrow \chi^m$

Output keys $\mathsf{pk} = (\mathbf{A}, \mathbf{x} = \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)$, $\mathsf{sk} = \mathbf{s}$



Linea

# Example: Regev Encryption [Regev05]

KeyGen

Sample uniform matrix $\mathbf{A}$, uniform $\mathbf{s}$ and $\mathbf{e} \leftarrow \chi^m$

Output keys pk $= (\mathbf{A}, \mathbf{x} = \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)$, sk $= \mathbf{s}$

$\text{Enc}_{\text{pk}}(\text{msg})$

Sample $\mathbf{z} \leftarrow \{-1,0,1\}^m$

$\mathbf{c}_0 = \mathbf{A}\mathbf{z}$, $c_1 = \mathbf{x}\mathbf{z} + \tau \cdot \text{msg}$

Output $(\mathbf{c}_0, c_1)$

Linea

# Example: Regev Encryption [Regev05]

KeyGen

Sample uniform matrix $\mathbf{A}$, uniform $\mathbf{s}$ and $\mathbf{e} \leftarrow \chi^m$

Output keys pk $= (\mathbf{A}, \mathbf{x} = \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)$, sk $= \mathbf{s}$

$\mathsf{Enc}_{\mathsf{pk}}(\mathsf{msg})$

Sample $\mathbf{z} \leftarrow \{-1,0,1\}^m$

$\mathbf{c}_0 = \mathbf{A}\mathbf{z}$, $c_1 = \mathbf{x}\mathbf{z} + \tau \cdot \mathsf{msg}$

Output $(\mathbf{c}_0, c_1)$

$\mathsf{Dec}_{\mathbf{sk}}(\mathsf{c}_0, \mathbf{c}_1)$

Compute $d = c_1 - \mathbf{s}^\top \mathbf{c}_0$.

Output $\gamma \in \mathbb{Z}_p$ s.t. $d - \tau \cdot \gamma \mod q$ closest to $0$

# Example: Regev Encryption [Regev05]

KeyGen

Sample uniform matrix $\mathbf{A}$, uniform $\mathbf{s}$ and $\mathbf{e} \leftarrow \chi^m$

Output keys pk $= (\mathbf{A}, \mathbf{x} = \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)$, sk $= \mathbf{s}$

Enc$_{\text{pk}}$(msg)

Sample $\mathbf{z} \leftarrow \{-1,0,1\}^m$

$\mathbf{c}_0 = \mathbf{A}\mathbf{z}$, $c_1 = \mathbf{x}\mathbf{z} + \tau \cdot \text{msg}$

Output $(\mathbf{c}_0, c_1)$

Dec$_{\mathbf{sk}}$($\mathrm{c}_0, \mathbf{c}_1$)

Compute $d = c_1 - \mathbf{s}^\top \mathbf{c}_0$.

Output $\gamma \in \mathbb{Z}_p$ s.t. $d - \tau \cdot \gamma \mod q$ closest to $0$

Compute $c_1 = \mathbf{x}\mathbf{z} + \tau \cdot \text{msg} - \mathbf{s}^\top \mathbf{A}\mathbf{z}$

$\mathbf{s}^\top \mathbf{A}\mathbf{z} + \mathbf{e}^\top \mathbf{z} + \tau \cdot \text{msg} - \mathbf{s}^\top \mathbf{A}\mathbf{z}$

# Example: Regev Encryption [Regev05]

KeyGen

Sample uniform matrix $\mathbf{A}$, uniform $\mathbf{s}$ and $\mathbf{e} \leftarrow \chi^m$

Output keys pk $= (\mathbf{A}, \mathbf{x} = \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)$, sk $= \mathbf{s}$

$\text{Enc}_{\text{pk}}(\text{msg})$

Sample $\mathbf{z} \leftarrow \{-1,0,1\}^m$

$\mathbf{c}_0 = \mathbf{A}\mathbf{z}$, $c_1 = \mathbf{x}\mathbf{z} + \tau \cdot \text{msg}$

Output $(\mathbf{c}_0, c_1)$

$\text{Dec}_{\mathbf{sk}}(\text{c}_0, \mathbf{c}_1)$

Compute $d = c_1 - \mathbf{s}^\top \mathbf{c}_0$.

Output $\gamma \in \mathbb{Z}_p$ s.t. $d - \tau \cdot \gamma \mod q$ closest to $0$

Compute $c_1 = \mathbf{x}\mathbf{z} + \tau \cdot \text{msg} - \mathbf{s}^\top \mathbf{A}\mathbf{z}$

$\mathbf{s}^\top \cancel{\mathbf{A}}\mathbf{z} + \mathbf{e}^\top \mathbf{z} + \tau \cdot \text{msg} - \mathbf{s}^\top \cancel{\mathbf{A}}\mathbf{z}$

# Example: Regev Encryption [Regev05]

KeyGen

Sample uniform matrix $\mathbf{A}$, uniform $\mathbf{s}$ and $\mathbf{e} \leftarrow \chi^m$

Output keys pk $= (\mathbf{A}, \mathbf{x} = \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)$, sk $= \mathbf{s}$

Enc$_{\mathsf{pk}}$(msg)

Sample $\mathbf{z} \leftarrow \{-1,0,1\}^m$

$\mathbf{c}_0 = \mathbf{A}\mathbf{z}$, $c_1 = \mathbf{x}\mathbf{z} + \tau \cdot \mathsf{msg}$

Output $(\mathbf{c}_0, c_1)$

Dec$_{\mathbf{sk}}(\mathsf{c}_0, \mathbf{c}_1)$

Compute $d = c_1 - \mathbf{s}^\top \mathbf{c}_0$.

Output $\gamma \in \mathbb{Z}_p$ s.t. $d - \tau \cdot \gamma \mod q$ closest to $0$

$\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ $\mathbf{s} \in \mathbb{Z}_q^n$ $\mathbf{x} = \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top \in \mathbb{Z}_q^m$

$\mathbf{z} \in \{-1,0,1\}^m$ $\mathsf{c}_0 \in \mathbb{Z}_q^n$ $\mathsf{c}_1 \in \mathbb{Z}_q$

# Example: Boyen's Signature [Boyen10]

KeyGen

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_{\mathbf{A}}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T}_{\mathbf{A}}$

# Example: Boyen's Signature [Boyen10]

KeyGen

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_{\mathbf{A}}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys $\mathsf{vk} = (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, $\mathsf{sk} = \mathbf{T}_{\mathbf{A}}$

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

Compute $\mathbf{C}_{\mathsf{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i \mathbf{C}_i$

Set $\mathbf{F}_{\mathbf{msg}} = [\mathbf{A} \,|\, \mathbf{C}_{\mathsf{msg}}]$

Use $\mathbf{T}_{\mathbf{A}}$ to generate short $\mathbf{d}$ such that $\mathbf{F}_{\mathsf{msg}} \cdot \mathbf{d} = \mathbf{0}$

Output $\mathbf{d}$ as the signature

Linea

# Example: Boyen's Signature [Boyen10]

**KeyGen**

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_\mathbf{A}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T}_\mathbf{A}$

**$\mathsf{Sign}_{\mathsf{sk}}(\mathsf{msg})$**

Compute $\mathbf{C}_{\mathsf{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i \mathbf{C}_i$

Set $\mathbf{F}_{\mathbf{msg}} = [\mathbf{A} \,|\, \mathbf{C}_{\mathsf{msg}}]$

Use $\mathbf{T}_\mathbf{A}$ to generate short $\mathbf{d}$ such that $\mathbf{F}_{\mathsf{msg}} \cdot \mathbf{d} = \mathbf{0}$

Output $\mathbf{d}$ as the signature

**$\mathsf{Verify}_{\mathsf{vk}}(\mathsf{msg}, \boldsymbol{\sigma})$**

Check that $\boldsymbol{\sigma}$ is short and non-zero.

Check that $\mathbf{F}_{\mathsf{msg}} \cdot \boldsymbol{\sigma} = \mathbf{0}$

Linea

# Example: Boyen's Signature [Boyen10]

KeyGen

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_{\mathbf{A}}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys $\mathsf{vk} = (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, $\mathsf{sk} = \mathbf{T}_{\mathbf{A}}$

$\mathbf{A} \in \mathbb{Z}_q^{n \times m}$

$\mathbf{F}_{\mathsf{msg}} \in \mathbb{Z}_q^{n \times 2m}$

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

Compute $\mathbf{C}_{\mathsf{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i \mathbf{C}_i$

Set $\mathbf{F}_{\mathsf{msg}} = [\mathbf{A} \,|\, \mathbf{C}_{\mathsf{msg}}]$

Use $\mathbf{T}_{\mathbf{A}}$ to generate short $\mathbf{d}$ such that $\mathbf{F}_{\mathsf{msg}} \cdot \mathbf{d} = \mathbf{0}$

Output $\mathbf{d}$ as the signature

$\mathsf{Verify}_{\mathsf{vk}}(\mathsf{msg}, \boldsymbol{\sigma})$

Check that $\boldsymbol{\sigma}$ is short and non-zero.

Check that $\mathbf{F}_{\mathsf{msg}} \cdot \boldsymbol{\sigma} = \mathbf{0}$

Linea

# Example: Encrypt the Signature

KeyGen

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_\mathbf{A}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T}_\mathbf{A}$

Regev Encryption

Output keys pk$' = (\mathbf{A}', \mathbf{x}' = \mathbf{s}'^\top \mathbf{A}' + \mathbf{e}'^\top)$, sk$' = \mathbf{s}'$

$\mathbf{c}_0' = \mathbf{A}'\mathbf{z}'$, $\mathbf{c}_1' = \mathbf{x}'\mathbf{z}' + \tau' \cdot \mathsf{msg}$

# Example: Encrypt the Signature

**KeyGen**

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_{\mathbf{A}}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T}_{\mathbf{A}}$

**Sign$_{\text{sk}}$(msg)**

Compute $\mathbf{C}_{\text{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \text{msg}_i \mathbf{C}_i$

Set $\mathbf{F}_{\mathbf{msg}} = [\mathbf{A} \,|\, \mathbf{C}_{\text{msg}}]$

Use $\mathbf{T}_{\mathbf{A}}$ to generate short $\mathbf{d}$ such that $\mathbf{F}_{\text{msg}} \cdot \mathbf{d} = \mathbf{0}$

Sample $\mathbf{Z}$, output $\mathbf{c}_0 = \mathbf{A}'\mathbf{Z}'$, $\mathbf{c}_1 = \mathbf{x}'\mathbf{Z}' + \tau' \cdot \mathbf{d}$

**Regev Encryption**

Output keys pk$' = (\mathbf{A}', \mathbf{x}' = \mathbf{s}^{\top}\mathbf{A}' + \mathbf{e}^{\top})$, sk$' = \mathbf{s}'$

$\mathbf{c}_0' = \mathbf{A}'\mathbf{z}'$, $\mathbf{c}_1' = \mathbf{x}'\mathbf{z}' + \tau' \cdot \text{msg}$

# Example: Encrypt the Signature

**KeyGen**

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T_A}$

Sample uniform $(\mathbf{C}_0 \dots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \dots \mathbf{C}_\ell)$, sk $= \mathbf{T_A}$

**$\mathsf{Sign}_{\mathsf{sk}}(\mathsf{msg})$**

Compute $\mathbf{C}_{\mathsf{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i \mathbf{C}_i$

Set $\mathbf{F_{msg}} = [\mathbf{A} \mid \mathbf{C}_{\mathsf{msg}}]$

Use $\mathbf{T_A}$ to generate short $\mathbf{d}$ such that $\mathbf{F}_{\mathsf{msg}} \cdot \mathbf{d} = \mathbf{0}$

Sample $\mathbf{Z}$, output $\mathbf{c}_0 = \mathbf{A}'\mathbf{Z}'$, $\mathbf{c}_1 = \mathbf{x}'\mathbf{Z}' + \tau' \cdot \mathbf{d}$

**$\mathsf{Verify}_{\mathsf{vk}}(\mathsf{msg}, \mathbf{c}_0, \mathbf{c}_1)$**

Check that $\mathbf{d}$ is short and non-zero.

Check that $\mathbf{F}_{\mathsf{msg}} \cdot \boldsymbol{\sigma} = \mathbf{0}$

**Regev Encryption**

Output keys pk$' = (\mathbf{A}', \mathbf{x}' = \mathbf{s}^\top \mathbf{A}' + \mathbf{e}^\top)$, sk$' = \mathbf{s}'$

$\mathbf{c}_0' = \mathbf{A}'\mathbf{z}'$, $\mathbf{c}_1' = \mathbf{x}'\mathbf{z}' + \tau' \cdot \mathsf{msg}$

Linea

# Example: Encrypt the Signature

**KeyGen**

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_\mathbf{A}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T}_\mathbf{A}$

**Sign$_{\text{sk}}$(msg)**

Compute $\mathbf{C}_{\text{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \text{msg}_i \mathbf{C}_i$

Set $\mathbf{F}_{\mathbf{msg}} = [\mathbf{A} \,|\, \mathbf{C}_{\text{msg}}]$

Use $\mathbf{T}_\mathbf{A}$ to generate short $\mathbf{d}$ such that $\mathbf{F}_{\text{msg}} \cdot \mathbf{d} = \mathbf{0}$

Sample $\mathbf{Z}$, output $\mathbf{c}_0 = \mathbf{A}'\mathbf{Z}'$, $\mathbf{c}_1 = \mathbf{x}'\mathbf{Z}' + \tau' \cdot \mathbf{d}$

**Verify$_{\text{vk}}$(msg, $\mathbf{c}_0$, $\mathbf{c}_1$)**

Check that $\mathbf{d}$ is short and non-zero.

Check that $\mathbf{F}_{\text{msg}} \cdot \boldsymbol{\sigma} = \mathbf{0}$

**Regev Encryption**

Output keys pk$' = (\mathbf{A}', \mathbf{x}' = \mathbf{s}^\top \mathbf{A}' + \mathbf{e}^\top)$, sk$' = \mathbf{s}'$

$\mathbf{c}_0' = \mathbf{A}'\mathbf{z}'$, $\mathbf{c}_1' = \mathbf{x}'\mathbf{z}' + \tau' \cdot \text{msg}$

we do not have $\mathbf{d}$ anymore!

Linea

# Example: Encrypt the Signature

**KeyGen**

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_{\mathbf{A}}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T}_{\mathbf{A}}$

**Sign$_{\mathsf{sk}}$(msg)**

Compute $\mathbf{C}_{\mathsf{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i \mathbf{C}_i$

Set $\mathbf{F}_{\mathbf{msg}} = [\mathbf{A} \,|\, \mathbf{C}_{\mathsf{msg}}]$

Use $\mathbf{T}_{\mathbf{A}}$ to generate short $\mathbf{d}$ such that $\mathbf{F}_{\mathsf{msg}} \cdot \mathbf{d} = \mathbf{0}$

Sample $\mathbf{Z}$, output $\mathbf{c}_0 = \mathbf{A}'\mathbf{Z}'$, $\mathbf{c}_1 = \mathbf{x}'\mathbf{Z}' + \tau' \cdot \mathbf{d}$ → generate proof $\pi$ and include it in the output.

**Verify$_{\mathsf{vk}}$(msg, $\mathbf{c}_0, \mathbf{c}_1$)**

Check that $\mathbf{d}$ is short and non-zero.

Check that $\mathbf{F}_{\mathsf{msg}} \cdot \boldsymbol{\sigma} = \mathbf{0}$

**Regev Encryption**

Output keys pk$' = (\mathbf{A}', \mathbf{x}' = \mathbf{s}'^{\top}\mathbf{A}' + \mathbf{e}'^{\top})$, sk$' = \mathbf{s}'$

$\mathbf{c}_0' = \mathbf{A}'\mathbf{z}'$, $\mathbf{c}_1' = \mathbf{x}'\mathbf{z}' + \tau' \cdot \mathsf{msg}$

# Example: Encrypt the Signature

KeyGen

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_{\mathbf{A}}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T}_{\mathbf{A}}$

$\text{Sign}_{\text{sk}}(\text{msg})$

Compute $\mathbf{C}_{\text{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \text{msg}_i \mathbf{C}_i$

Set $\mathbf{F}_{\mathbf{msg}} = [\mathbf{A} \,|\, \mathbf{C}_{\text{msg}}]$

Use $\mathbf{T}_{\mathbf{A}}$ to generate short $\mathbf{d}$ such that $\mathbf{F}_{\text{msg}} \cdot \mathbf{d} = \mathbf{0}$

Sample $\mathbf{Z}$, output $\mathbf{c}_0 = \mathbf{A}'\mathbf{Z}'$, $\mathbf{c}_1 = \mathbf{x}'\mathbf{Z}' + \tau' \cdot \mathbf{d}$ → generate proof $\pi$ and include it in the output.

$\text{Verify}_{\text{vk}}(\text{msg}, \mathbf{c}_0, \mathbf{c}_1)$

Check that $\mathbf{d}$ is short and non-zero. → We also need proofs for the shortness and non-zero checks.

Check that $\mathbf{F}_{\text{msg}} \cdot \boldsymbol{\sigma} = \mathbf{0}$

Regev Encryption

Output keys pk$' = (\mathbf{A}', \mathbf{x}' = \mathbf{s}^\top \mathbf{A}' + \mathbf{e}^\top)$, sk$' = \mathbf{s}'$

$\mathbf{c}_0' = \mathbf{A}'\mathbf{z}'$, $\mathbf{c}_1' = \mathbf{x}'\mathbf{z}' + \tau' \cdot \text{msg}$

Linea

# Example: Encrypt the Signature

**KeyGen**

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_\mathbf{A}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T}_\mathbf{A}$

**Regev Encryption**

Output keys pk$' = (\mathbf{A}', \mathbf{x}' = \mathbf{s}'^\top \mathbf{A}' + \mathbf{e}'^\top)$, sk$' = \mathbf{s}'$

$\mathbf{c}_0' = \mathbf{A}'\mathbf{z}'$, $\mathbf{c}_1' = \mathbf{x}'\mathbf{z}' + \tau' \cdot \mathsf{msg}$

**Sign$_{\mathsf{sk}}$(msg)**

Compute $\mathbf{C}_{\mathsf{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i \mathbf{C}_i$

Set $\mathbf{F}_{\mathbf{msg}} = [\mathbf{A} \mid \mathbf{C}_{\mathsf{msg}}]$

Use $\mathbf{T}_\mathbf{A}$ to generate short $\mathbf{d}$ such that $\mathbf{F}_{\mathsf{msg}} \cdot \mathbf{d} = \mathbf{0}$

Sample $\mathbf{Z}$, output $\mathbf{c}_0 = \mathbf{A}'\mathbf{Z}'$, $\mathbf{c}_1 = \mathbf{x}'\mathbf{Z}' + \tau' \cdot \mathbf{d}$ → generate proof $\pi$ and include it in the output.

**Verify$_{\mathsf{vk}}$(msg, $\mathbf{c}_0, \mathbf{c}_1$)**

Check that $\mathbf{d}$ is short and non-zero. → We also need proofs for the shortness and non-zero checks.

Check that $\mathbf{F}_{\mathsf{msg}} \cdot \boldsymbol{\sigma} = \mathbf{0}$ → homomorphic evaluation leads to an encryption of $\mathbf{0}$

check it using a NIZK proof $\pi$

Linea

# Example: Boyen's Signature [Boyen10]

**KeyGen**

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_\mathbf{A}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T}_\mathbf{A}$

**$\mathsf{Sign}_{\mathsf{sk}}(\mathsf{msg})$**

Compute $\mathbf{C}_{\mathsf{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i \mathbf{C}_i$

Set $\mathbf{F}_{\mathbf{msg}} = [\mathbf{A} \,|\, \mathbf{C}_{\mathsf{msg}}]$

Use $\mathbf{T}_\mathbf{A}$ to generate short $\mathbf{d}$ such that $\mathbf{F}_{\mathsf{msg}} \cdot \mathbf{d} = \mathbf{0}$

Sample $\mathbf{Z}$, output $\mathbf{c}_0 = \mathbf{A}'\mathbf{Z}'$, $\mathbf{c}_1 = \mathbf{x}'\mathbf{Z}' + \tau' \cdot \mathbf{d}$

**$\mathsf{Verify}_{\mathsf{vk}}(\mathsf{msg}, \mathbf{c}_0, \mathbf{c}_1)$**

Check that $\mathbf{d}$ is short and non-zero.

Check that $\mathbf{F}_{\mathsf{msg}} \cdot \boldsymbol{\sigma} = \mathbf{0}$

**Regev Encryption**

Output keys pk$' = (\mathbf{A}', \mathbf{x}' = \mathbf{s}^\top \mathbf{A}' + \mathbf{e}^\top)$, sk$' = \mathbf{s}'$

$\mathbf{c}_0' = \mathbf{A}'\mathbf{z}'$, $\mathbf{c}_1' = \mathbf{x}'\mathbf{z}' + \tau' \cdot \mathsf{msg}$

Wanted: a proof that something encrypted is:

- short
- non-zero
- an encryption of $\mathbf{0}$

# Example: Boyen's Signature [Boyen10]

**KeyGen**

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_\mathbf{A}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T}_\mathbf{A}$

**Sign$_{\text{sk}}$(msg)**

Compute $\mathbf{C}_{\text{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \text{msg}_i \mathbf{C}_i$

Set $\mathbf{F}_{\mathbf{msg}} = [\mathbf{A} \,|\, \mathbf{C}_{\text{msg}}]$

Use $\mathbf{T}_\mathbf{A}$ to generate short $\mathbf{d}$ such that $\mathbf{F}_{\text{msg}} \cdot \mathbf{d} = \mathbf{0}$

Sample $\mathbf{Z}$, output $\mathbf{c}_0 = \mathbf{A}'\mathbf{Z}'$, $\mathbf{c}_1 = \mathbf{x}'\mathbf{Z}' + \tau' \cdot \mathbf{d}$

**Verify$_{\text{vk}}$(msg, $\mathbf{c}_0, \mathbf{c}_1$)**

Check that $\mathbf{d}$ is short and non-zero.

Check that $\mathbf{F}_{\text{msg}} \cdot \boldsymbol{\sigma} = \mathbf{0}$

**Regev Encryption**

Output keys pk$' = (\mathbf{A}', \mathbf{x}' = \mathbf{s}^\top \mathbf{A}' + \mathbf{e}^\top)$, sk$' = \mathbf{s}'$

$\mathbf{c}_0' = \mathbf{A}'\mathbf{z}'$, $\mathbf{c}_1' = \mathbf{x}'\mathbf{z}' + \tau' \cdot \text{msg}$

Wanted: a proof that something encrypted is:

- short
- non-zero
- an encryption of $\mathbf{0}$

Wanted: the encrypted signature becomes hidden.

Goal:
- hide something short (smudge it)
- despite being hidden, we can check that the original was short.

# Example: Boyen's Signature [Boyen10]

KeyGen

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T_A}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T_A}$

$\text{Sign}_{sk}(\text{msg})$

Compute $\mathbf{C}_{msg} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \text{msg}_i \mathbf{C}_i$

Set $\mathbf{F_{msg}} = [\mathbf{A} \mid \mathbf{C}_{msg}]$

Use $\mathbf{T_A}$ to generate short $\mathbf{d}$ such that $\mathbf{F}_{msg} \cdot \mathbf{d} = \mathbf{0}$

Sample $\mathbf{Z}$, output $\mathbf{c}_0 = \mathbf{A}'\mathbf{Z}'$, $\mathbf{c}_1 = \mathbf{x}'\mathbf{Z}' + \tau' \cdot \mathbf{d}$

$\text{Verify}_{vk}(\text{msg}, \mathbf{c}_0, \mathbf{c}_1)$

Check that $\mathbf{d}$ is short and non-zero.

Check that $\mathbf{F}_{msg} \cdot \boldsymbol{\sigma} = \mathbf{0}$

Regev Encryption

Output keys $\text{pk}' = (\mathbf{A}', \mathbf{x}' = \mathbf{s}^\top \mathbf{A}' + \mathbf{e}^\top)$, $\text{sk}' = \mathbf{s}'$

$\mathbf{c}_0' = \mathbf{A}'\mathbf{z}'$, $\mathbf{c}_1' = \mathbf{x}'\mathbf{z}' + \tau' \cdot \text{msg}$

Wanted: a proof that something encrypted is:

- short
- non-zero
- an encryption of $\mathbf{0}$

Wanted: the encrypted signature becomes hidden.

Goal:
- hide something short (smudge it)
- despite being hidden, we can check that the original was short.

Smudging is not with superpolynomially larger noise.

# Structure-Preserving Sets, First Step

Goal: hide the short signatures

Set $S \subseteq \mathbb{Z}_q^d$ is structure-preserving if there exists a noise distribution $D$ such that:

- We can hide the elements of $S$ using smudging.

# Structure-Preserving Sets, First Step

Set $S \subseteq \mathbb{Z}_q^d$ is structure-preserving if there exists a noise distribution $D$ such that:

- We can hide the elements of $S$ using smudging.

- We can still check whether smudged elements belong to the original set $S$

# Structure-Preserving Sets, First Step

Set $S \subseteq \mathbb{Z}_q^d$ is structure-preserving if there exists a noise distribution $D$ such that:

- $D$ smudges the elements of $S$, for any $s, s' \in S$ and any $d \in D$:

$$s + d \approx_S s' + d$$

- $D$ smudging preserves membership and non-membership in $S$:

$$S + \mathrm{supp}(D) \cap (\mathbb{Z}_p \backslash S) + \mathrm{supp}(D) = \varnothing$$

# Structure-Preserving Sets, First Step

Set $S \subseteq \mathbb{Z}_q^d$ is structure-preserving if there exists a noise distribution $D$ such that:

- $D$ smudges the elements of $S$, for any $s, s' \in S$ and any $d \in D$:

$$s + d \approx_S s' + d$$

- $D$ smudging preserves membership and non-membership in $S$:

$$S + \text{supp}(D) \cap (\mathbb{Z}_p \backslash S) + \text{supp}(D) = \varnothing$$

We will be able to check if the hidden signature is short (membership check).

Linea

# Structure-Preserving Sets, First Step

Smudging will also require rejection sampling.

# Structure-Preserving Sets

Set $S \subseteq \mathbb{Z}_q^d$ is structure-preserving if there exists a noise distribution $D$, constant $\alpha$, and function success s.t.

- $D$ smudges the elements of $S$, for any $s, s' \in S$ and any $d \in D$:

# Structure-Preserving Sets

Set $S \subseteq \mathbb{Z}_q^d$ is structure-preserving if there exists a noise distribution $D$, constant $\alpha$, and function success s.t.

- $D$ smudges the elements of $S$, for any $s, s' \in S$ and any $d \in D$:

| $d \leftarrow_r D$<br>Output $s + d$ with probability $\text{success}(s, s', d)$<br>$\perp$ with $1 - \text{success}(s, s', d)$ | $\approx_S$ | $d \leftarrow_r D$<br>Output $s' + d$ with probability $\alpha$<br>$\perp$ with probability $1 - \alpha$ |
| --- | --- | --- |

Linea

# Structure-Preserving Sets

Set $S \subseteq \mathbb{Z}_q^d$ is structure-preserving if there exists a noise distribution $D$, constant $\alpha$, and function success s.t.

- $D$ smudges the elements of $S$, for any $s, s' \in S$ and any $d \in D$:

| | | |
|---|---|---|
| $d \leftarrow_r D$<br>Output $s + d$ with probability $\mathrm{success}(s, s', d)$<br>$\perp$ with $1 - \mathrm{success}(s, s', d)$ | $\approx_S$ | $d \leftarrow_r D$<br>Output $s' + d$ with probability $\alpha$<br>$\perp$ with probability $1 - \alpha$ |

success and $\alpha$ stem from [Lyubashevsky12]'s rejection sampling.

# Structure-Preserving Sets

Set $S \subseteq \mathbb{Z}_q^d$ is structure-preserving if there exists a noise distribution $D$, constant $\alpha$, and function success s.t.

- $D$ smudges the elements of $S$, for any $s, s' \in S$ and any $d \in D$:

| | | |
|---|---|---|
| $d \leftarrow_r D$<br>Output $s + d$ with probability success$(s, s', d)$<br>$\perp$ with $1 -$ success$(s, s', d)$ | $\approx_S$ | $d \leftarrow_r D$<br>Output $s' + d$ with probability $\alpha$<br>$\perp$ with probability $1 - \alpha$ |

success and $\alpha$ stem from [Lyubashevsky12]'s rejection sampling.

- Membership for $D$ and $(S + D)$ are easy.

$$\boxed{S + D} \cap \boxed{\left(\mathbb{Z}_q^d \backslash S\right) + D} = \varnothing$$

Linea

# Structure-Preserving Sets

Set $S \subseteq \mathbb{Z}_q^d$ is structure-preserving if there exists a noise distribution $D$, constant $\alpha$, and function success s.t.

- $D$ smudges the elements of $S$, for any $s, s' \in S$ and any $d \in D$:

| | | |
|---|---|---|
| $d \leftarrow_r D$<br>Output $s + d$ with probability success$(s, s', d)$<br>$\perp$ with $1 -$ success$(s, s', d)$ | $\approx_S$ | $d \leftarrow_r D$<br>Output $s' + d$ with probability $\alpha$<br>$\perp$ with probability $1 - \alpha$ |

success and $\alpha$ stem from [Lyubashevsky12]'s rejection sampling.

- Membership for $D$ and $(S + D)$ are easy.

$$\boxed{S + D} \cap \boxed{\left(\mathbb{Z}_q^d \backslash S\right) + D} \approx \varnothing$$

noisiness around $S$, actual definition w.r.t. $B_\delta(S)$ and noise $\delta$

# Structure-Preserving Set—Example

Example: any coset of any additive subgroup $G \subseteq \mathbb{Z}_q^d$.

Example: any singleton (as a coset of the additive group $\{\mathbf{0}\}$)

Example: Every set $S$ where $S - S \in B_T(\{0\})$

meaning that the vectors are close to each other

Example: if $S_1$ and $S_2$ are structure preserving, so is $S_1 \times S_2$

Linea

# Example: Boyen's Signature [Boyen10]

KeyGen

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_{\mathbf{A}}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T}_{\mathbf{A}}$

$\mathbf{A} \in \mathbb{Z}_q^{n \times m}$

$\mathbf{F}_{\text{msg}} \in \mathbb{Z}_q^{n \times 2m}$

$\text{Sign}_{\text{sk}}(\text{msg})$

Compute $\mathbf{C}_{\text{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \text{msg}_i \mathbf{C}_i$

Set $\mathbf{F}_{\mathbf{msg}} = [\mathbf{A} \,|\, \mathbf{C}_{\text{msg}}]$

Use $\mathbf{T}_{\mathbf{A}}$ to generate short $\mathbf{d}$ such that $\mathbf{F}_{\text{msg}} \cdot \mathbf{d} = \mathbf{0}$

Output $\mathbf{d}$ as the signature

$\text{Verify}_{\text{vk}}(\text{msg}, \boldsymbol{\sigma})$

Check that $\boldsymbol{\sigma}$ is short and non-zero.

Check that $\mathbf{F}_{\text{msg}} \cdot \boldsymbol{\sigma} = \mathbf{0}$

$\text{Verify}_{\text{vk}}(\text{msg}, \boldsymbol{\sigma})$

Check that $\sigma$ belongs to structure preserving set of short vectors.

Check that $\mathbf{F}_{\text{msg}} \cdot \boldsymbol{\sigma}$ belongs to the structure preserving set $\{\mathbf{0}\}$

We ignore non-zero check for now

# Example: Boyen's Signature [Boyen10]

**KeyGen**

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T_A}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T_A}$

$\mathbf{A} \in \mathbb{Z}_q^{n \times m}$

$\mathbf{F}_{\text{msg}} \in \mathbb{Z}_q^{n \times 2m}$

**$\text{Sign}_{\text{sk}}(\text{msg})$**

Compute $\mathbf{C}_{\text{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \text{msg}_i \mathbf{C}_i$

Set $\mathbf{F_{msg}} = [\mathbf{A} \,|\, \mathbf{C}_{\text{msg}}]$

Use $\mathbf{T_A}$ to generate short $\mathbf{d}$ such that $\mathbf{F}_{\text{msg}} \cdot \mathbf{d} = \mathbf{0}$

Output $\mathbf{d}$ as the signature

**$\text{Verify}_{\text{vk}}(\text{msg}, \boldsymbol{\sigma})$**

Check that $\boldsymbol{\sigma}$ is short and non-zero.

Check that $\mathbf{F}_{\text{msg}} \cdot \boldsymbol{\sigma} = \mathbf{0}$

**$\text{Verify}_{\text{vk}}(\text{msg}, \boldsymbol{\sigma})$**

Let $f^1_{\text{vk,msg}}(\sigma) = \mathbf{F}_{\text{msg}} \cdot \sigma,\ f^2_{\text{vk,msg}}(\sigma) = \sigma$

Consider structure-preserving sets $S_1 = \{\mathbf{0}\}$ and $S_2$, a ball of short vectors.

Output 1 if $f^i_{\text{vk,msg}}(\sigma) = S_i$ for both $i = \{1,2\}$



Linea

# Example: Boyen's Signature [Boyen10]

**KeyGen**

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T_A}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T_A}$

$\mathbf{A} \in \mathbb{Z}_q^{n \times m}$

$\mathbf{F}_{\mathsf{msg}} \in \mathbb{Z}_q^{n \times 2m}$

**$\mathsf{Sign}_{\mathsf{sk}}(\mathsf{msg})$**

Compute $\mathbf{C}_{\mathsf{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i \mathbf{C}_i$

Set $\mathbf{F_{msg}} = [\mathbf{A} \,|\, \mathbf{C}_{\mathsf{msg}}]$

Use $\mathbf{T_A}$ to generate short $\mathbf{d}$ such that $\mathbf{F}_{\mathsf{msg}} \cdot \mathbf{d} = \mathbf{0}$

Output $\mathbf{d}$ as the signature

**$\mathsf{Verify}_{\mathsf{vk}}(\mathsf{msg}, \boldsymbol{\sigma})$**

Let $f^1_{\mathsf{vk,msg}}(\sigma) = \mathbf{F}_{\mathsf{msg}} \cdot \sigma$, $f^2_{\mathsf{vk,msg}}(\sigma) = \sigma$

Consider structure-preserving sets $S_1 = \{\mathbf{0}\}$ and $S_2$, a ball of short vectors.

Output 1 if $f^i_{\mathsf{vk,msg}}(\sigma) = S_i$ for both $i = \{1,2\}$

**$\mathsf{Verify}_{\mathsf{vk}}(\mathsf{msg}, \boldsymbol{\sigma})$**

Check that $\boldsymbol{\sigma}$ is short and non-zero. ← we ignore the non-zero check for now

Check that $\mathbf{F}_{\mathsf{msg}} \cdot \boldsymbol{\sigma} = \mathbf{0}$

Linea

# Structure-Preserving Signature Definition

A structure-preserving signature for a function family $\mathscr{F}$ is a digital signature where for all verification keys $\mathsf{vk}$, message $\mathsf{msg}$ and signature $\sigma$,

$$\mathsf{Verify}(\mathsf{vk}, \mathsf{msg}, \sigma) = 1 \iff f_{\mathsf{vk},\mathsf{msg}}(\sigma) \in S$$

$f$ depends on $\mathsf{vk}$ and $\mathsf{msg}$

structure-preserving set S

# Structure-Preserving Signature Definition

A structure-preserving signature for a function family $\mathscr{F}$ is a digital signature where for all verification keys $\mathsf{vk}$, message $\mathsf{msg}$ and signature $\sigma$,

$$\mathsf{Verify}(\mathsf{vk}, \mathsf{msg}, \sigma) = 1 \iff f_{\mathsf{vk},\mathsf{msg}}(\sigma) \in S$$

$f$ depends on $\mathsf{vk}$ and $\mathsf{msg}$ structure-preserving set S

The actual definition is more general to cover strongly unforgeable schemes.

It applies to [Boyen10], [Rückert10] and a new Inhomogenous SIS-based scheme we introduce in this paper.

modification of [Rúckert10] with delegation strategy of [ABB10]

Linea

# Formalising SPS Encryption from Regev Encryption

KeyGen

Sample uniform matrix $\mathbf{A}$, uniform $\mathbf{s}$ and $\mathbf{e} \leftarrow \chi^m$

Output keys pk $= (\mathbf{A}, \mathbf{x} = \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)$, sk $= \mathbf{s}$

Enc$_{\mathsf{pk}}$(msg)

Sample $\mathbf{z} \leftarrow \{-1,0,1\}^m$

$\mathbf{c}_0 = \mathbf{A}\mathbf{z}$, $c_1 = \mathbf{x}\mathbf{z} + \tau \cdot \mathsf{msg}$

Output $(\mathbf{c}_0, c_1)$

Dec$_{\mathbf{sk}}(\mathrm{c}_0, \mathbf{c}_1)$

Compute $d = c_1 - \mathbf{s}^\top \mathbf{c}_0$.

Output $\gamma \in \mathbb{Z}_p$ s.t. $d - \tau \cdot \gamma \mod q$ closest to $0$

KeyGen

Matrix $\mathbf{B} \in \mathbb{Z}_q^{d \times \tau}$

# Formalising SPS Encryption from Regev Encryption

**KeyGen**

Sample uniform matrix $\mathbf{A}$, uniform $\mathbf{s}$ and $\mathbf{e} \leftarrow \chi^m$

Output keys $\mathsf{pk} = (\mathbf{A}, \mathbf{x} = \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)$, $\mathsf{sk} = \mathbf{s}$

**$\mathsf{Enc}_{\mathsf{pk}}(\mathsf{msg})$**

Sample $\mathbf{z} \leftarrow \{-1,0,1\}^m$

$\mathbf{c}_0 = \mathbf{A}\mathbf{z}$, $c_1 = \mathbf{x}\mathbf{z} + \tau \cdot \mathsf{msg}$

Output $(\mathbf{c}_0, c_1)$

**$\mathsf{Dec}_{\mathbf{sk}}(\mathsf{c}_0, \mathbf{c}_1)$**

Compute $d = c_1 - \mathbf{s}^\top \mathbf{c}_0$.

Output $\gamma \in \mathbb{Z}_p$ s.t. $d - \tau \cdot \gamma \mod q$ closest to $0$

**KeyGen**

Matrix $\mathbf{B} \in \mathbb{Z}_q^{d \times \tau}$

**Enc**

invertible additive homomorphic encoding $g : \mathscr{M} \rightarrow \mathbb{Z}_q^d$

Sample randomness $\mathbf{r} \leftarrow_r \mathscr{R}$

Ciphertext will be $\mathsf{ct} = \mathbf{B} \cdot \mathbf{r} + g(\mathsf{msg})$

Linea

# Formalising SPS Encryption from Regev Encryption

**KeyGen**

Sample uniform matrix $\mathbf{A}$, uniform $\mathbf{s}$ and $\mathbf{e} \leftarrow \chi^m$

Output keys $\mathsf{pk} = (\mathbf{A}, \mathbf{x} = \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)$, $\mathsf{sk} = \mathbf{s}$

**$\mathsf{Enc}_{\mathsf{pk}}(\mathsf{msg})$**

Sample $\mathbf{z} \leftarrow \{-1,0,1\}^m$

$\mathbf{c}_0 = \mathbf{A}\mathbf{z}$, $c_1 = \mathbf{x}\mathbf{z} + \tau \cdot \mathsf{msg}$

Output $(\mathbf{c}_0, c_1)$

**$\mathsf{Dec}_{\mathbf{sk}}(\mathsf{c}_0, \mathbf{c}_1)$**

Compute $d = c_1 - \mathbf{s}^\top \mathbf{c}_0$.

Output $\gamma \in \mathbb{Z}_p$ s.t. $d - \tau \cdot \gamma \mod q$ closest to $0$

**KeyGen**

Matrix $\mathbf{B} \in \mathbb{Z}_q^{d \times \tau}$

**Enc**

invertible additive homomorphic encoding $g : \mathscr{M} \to \mathbb{Z}_q^d$

Sample randomness $\mathbf{r} \leftarrow_r \mathscr{R}$

Ciphertext will be $\mathsf{ct} = \mathbf{B} \cdot \mathbf{r} + g(\mathsf{msg})$

Matrix $\mathbf{B} = \mathbf{I}_n \otimes \begin{pmatrix} \mathbf{A} \\ \mathbf{X} \end{pmatrix}$

Matrix $g(\mathsf{msg}_1 \ldots \mathsf{msg}_\alpha) = \mathbf{I}_n \otimes \begin{pmatrix} \mathbf{0} \\ \tau \cdot \mathsf{msg}_1 \\ \vdots \\ \mathbf{0} \\ \tau \cdot \mathsf{msg}_\alpha \end{pmatrix}$

Linea

# Formalising SPS Encryption from Regev Encryption

**KeyGen**

Sample uniform matrix $\mathbf{A}$, uniform $\mathbf{s}$ and $\mathbf{e} \leftarrow \chi^m$

Output keys $\mathsf{pk} = (\mathbf{A}, \mathbf{x} = \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)$, $\mathsf{sk} = \mathbf{s}$

**$\mathsf{Enc}_{\mathsf{pk}}(\mathsf{msg})$**

Sample $\mathbf{z} \leftarrow \{-1,0,1\}^m$

$\mathbf{c}_0 = \mathbf{A}\mathbf{z}$, $c_1 = \mathbf{x}\mathbf{z} + \tau \cdot \mathsf{msg}$

Output $(\mathbf{c}_0, c_1)$

**$\mathsf{Dec}_{\mathbf{sk}}(\mathbf{c}_0, \mathbf{c}_1)$**

Compute $d = c_1 - \mathbf{s}^\top \mathbf{c}_0$.

Output $\gamma \in \mathbb{Z}_p$ s.t. $d - \tau \cdot \gamma \mod q$ closest to $0$

**KeyGen**

Matrix $\mathbf{B} \in \mathbb{Z}_q^{d \times \tau}$

**Enc**

invertible additive homomorphic encoding $g : \mathscr{M} \to \mathbb{Z}_q^d$

Sample randomness $\mathbf{r} \leftarrow_r \mathscr{R}$

Ciphertext will be $\mathsf{ct} = \mathbf{B} \cdot \mathbf{r} + g(\mathsf{msg})$

$\mathbf{r}$ belongs to a structure-preserving set $R$ with overwhelming probability.
this also models Gaussian noise like in dual Regev.

Linea

# Formalising SPS Encryption from Regev Encryption

**KeyGen**

Sample uniform matrix $\mathbf{A}$, uniform $\mathbf{s}$ and $\mathbf{e} \leftarrow \chi^m$

Output keys $\mathsf{pk} = (\mathbf{A}, \mathbf{x} = \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)$, $\mathsf{sk} = \mathbf{s}$

**$\mathsf{Enc}_{\mathsf{pk}}(\mathsf{msg})$**

Sample $\mathbf{z} \leftarrow \{-1,0,1\}^m$

$\mathbf{c}_0 = \mathbf{A}\mathbf{z}$, $c_1 = \mathbf{x}\mathbf{z} + \tau \cdot \mathsf{msg}$

Output $(\mathbf{c}_0, c_1)$

**$\mathsf{Dec}_{\mathbf{sk}}(\mathsf{c}_0, \mathbf{c}_1)$**

Compute $d = c_1 - \mathbf{s}^\top \mathbf{c}_0$.

Output $\gamma \in \mathbb{Z}_p$ s.t. $d - \tau \cdot \gamma \mod q$ closest to $0$

**KeyGen**

Matrix $\mathbf{B} \in \mathbb{Z}_q^{d\times\tau}$

**Enc**

invertible additive homomorphic encoding $g : \mathscr{M} \to \mathbb{Z}_q^d$

Sample randomness $\mathbf{r} \leftarrow_r \mathscr{R}$

Ciphertext will be $\mathsf{ct} = \mathbf{B} \cdot \mathbf{r} + g(\mathsf{msg})$

$\mathbf{r}$ belongs to a structure-preserving set $R$ with overwhelming probability.
this also models Gaussian noise like in dual Regev.

allows for message homomorphism



Linea

# Formalising SPS Encryption from Regev Encryption

KeyGen

Sample uniform matrix $\mathbf{A}$, uniform $\mathbf{s}$ and $\mathbf{e} \leftarrow \chi^m$

Output keys pk $= (\mathbf{A}, \mathbf{x} = \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)$, sk $= \mathbf{s}$

Enc$_{\text{pk}}$(msg)

Sample $\mathbf{z} \leftarrow \{-1,0,1\}^m$

$\mathbf{c}_0 = \mathbf{A}\mathbf{z}$, $c_1 = \mathbf{x}\mathbf{z} + \tau \cdot \text{msg}$

Output $(\mathbf{c}_0, c_1)$

Dec$_{\mathbf{sk}}(\mathrm{c}_0, \mathbf{c}_1)$

Compute $d = c_1 - \mathbf{s}^\top \mathbf{c}_0$.

Output $\gamma \in \mathbb{Z}_p$ s.t. $d - \tau \cdot \gamma \mod q$ closest to $0$

KeyGen

Matrix $\mathbf{B} \in \mathbb{Z}_q^{d \times \tau}$

Enc

invertible additive homomorphic encoding $g : \mathscr{M} \to \mathbb{Z}_q^d$

Sample randomness $\mathbf{r} \leftarrow_r \mathscr{R}$

Ciphertext will be ct $= \mathbf{B} \cdot \mathbf{r} + g(\text{msg})$

$\mathbf{r}$ belongs to a structure-preserving set $R$ with overwhelming probability.
this also models Gaussian noise like in dual Regev.

allows for message homomorphism

the actual definition also covers a series of noise properties.

Linea

# Structure-Preserving Encryption Definition

In a structure-preserving encryption scheme, the public key is expressible as a matrix $\mathbf{B}$.

The randomness space $R$ is a structure preserving set. and $g$ is a invertible additive homomorphism.

$$\mathsf{Enc}(\mathsf{pk}, \mathsf{msg}; \mathbf{r}) \quad = \quad \mathbf{B}\mathbf{r} + g(\mathsf{msg})$$

# Regev Encryption of a Boyen Signature

KeyGen

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_\mathbf{A}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T}_\mathbf{A}$

$\text{Sign}_{\text{sk}}(\text{msg})$

Compute $\mathbf{C}_{\text{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \text{msg}_i \mathbf{C}_i$

Set $\mathbf{F}_{\mathbf{msg}} = [\mathbf{A} \,|\, \mathbf{C}_{\text{msg}}]$

Use $\mathbf{T}_\mathbf{A}$ to generate short $\boldsymbol{\sigma}$ such that $\mathbf{F}_{\text{msg}} \cdot \boldsymbol{\sigma} = \mathbf{0}$

Sample $\mathbf{r}$, output $\boldsymbol{\sigma}^{\text{enc}} = \mathbf{B} \cdot \mathbf{r} + g(\mathbf{d})$

$\text{Verify}_{\text{vk}}(\text{msg}, \boldsymbol{\sigma})$

Apply $f^i_{\text{vk,msg}}$ homomorphically on $\boldsymbol{\sigma}^{\text{enc}}$

Get $\boldsymbol{\sigma}_i^{\text{enc}} = \mathbf{B} \cdot \mathbf{r}_i + g\left(f^i_{\text{vk,msg}}(\mathbf{d})\right)$

We need a way to check that $f^i_{\text{vk,msg}}(\mathbf{d}) \in S_i$

Regev SPS Encryption

Public-key matrix $\mathbf{B} \in \mathbb{Z}_q^{d \times \tau}$

invertible additive homomorphic encoding $g : \mathscr{M} \to \mathbb{Z}_q^d$

Sample randomness $\mathbf{r} \leftarrow_r \mathscr{R}$

Ciphertext will be ct $= \mathbf{B} \cdot \mathbf{r} + g(\text{msg})$

$\text{Verify}_{\text{vk}}(\text{msg}, \boldsymbol{\sigma})$ for Boyen SPS

Let $f^1_{\text{vk,msg}}(\boldsymbol{\sigma}) = \mathbf{F}_{\text{msg}} \cdot \boldsymbol{\sigma}$, $f^2_{\text{vk,msg}}(\boldsymbol{\sigma}) = \boldsymbol{\sigma}$

Consider structure-preserving sets $S_1 = \{\mathbf{0}\}$

and $S_2$, a ball of short vectors.

Output 1 if $f^i_{\text{vk,msg}}(\boldsymbol{\sigma}) = S_i$ for both $i = \{1,2\}$

computable since $g$ is homomorphic

# Regev Encryption of a Boyen Signature

**KeyGen**

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_{\mathbf{A}}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T}_{\mathbf{A}}$

**Sign$_{\text{sk}}$(msg)**

Compute $\mathbf{C}_{\text{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \text{msg}_i \mathbf{C}_i$

Set $\mathbf{F}_{\mathbf{msg}} = [\mathbf{A} \,|\, \mathbf{C}_{\text{msg}}]$

Use $\mathbf{T}_{\mathbf{A}}$ to generate short $\boldsymbol{\sigma}$ such that $\mathbf{F}_{\text{msg}} \cdot \boldsymbol{\sigma} = \mathbf{0}$

Sample $\mathbf{r}$, output $\boldsymbol{\sigma}^{\text{enc}} = \mathbf{B} \cdot \mathbf{r} + g(\mathbf{d})$

**Verify$_{\text{vk}}$(msg, $\boldsymbol{\sigma}$)**

Apply $f^i_{\text{vk,msg}}$ homomorphically on $\boldsymbol{\sigma}^{\text{enc}}$

Get $\boldsymbol{\sigma}_i^{\text{enc}} = \mathbf{B} \cdot \mathbf{r}_i + g\left(f^i_{\text{vk,msg}}(\mathbf{d})\right)$

We need a way to check that $f^i_{\text{vk,msg}}(\mathbf{d}) \in S_i$

**Regev SPS Encryption**

Public-key matrix $\mathbf{B} \in \mathbb{Z}_q^{d \times \tau}$

invertible additive homomorphic encoding $g : \mathscr{M} \to \mathbb{Z}_q^d$

Sample randomness $\mathbf{r} \leftarrow_r \mathscr{R}$

Ciphertext will be ct $= \mathbf{B} \cdot \mathbf{r} + g(\text{msg})$

**Verify$_{\text{vk}}$(msg, $\boldsymbol{\sigma}$) for Boyen SPS**

Let $f^1_{\text{vk,msg}}(\boldsymbol{\sigma}) = \mathbf{F}_{\text{msg}} \cdot \boldsymbol{\sigma}$, $f^2_{\text{vk,msg}}(\boldsymbol{\sigma}) = \boldsymbol{\sigma}$

Consider structure-preserving sets $S_1 = \{\mathbf{0}\}$

and $S_2$, a ball of short vectors.

Output 1 if $f^i_{\text{vk,msg}}(\boldsymbol{\sigma}) = S_i$ for both $i = \{1,2\}$

computable since $g$ is homomorphic

How to do this?



Linea

# Regev Encryption of a Boyen Signature

**KeyGen**

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_\mathbf{A}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys $\mathsf{vk} = (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, $\mathsf{sk} = \mathbf{T}_\mathbf{A}$

**Sign$_{\mathsf{sk}}$(msg)**

Compute $\mathbf{C}_{\mathsf{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i \mathbf{C}_i$

Set $\mathbf{F}_{\mathbf{msg}} = [\mathbf{A} \,|\, \mathbf{C}_{\mathsf{msg}}]$

Use $\mathbf{T}_\mathbf{A}$ to generate short $\boldsymbol{\sigma}$ such that $\mathbf{F}_{\mathsf{msg}} \cdot \boldsymbol{\sigma} = \mathbf{0}$

Sample $\mathbf{r}$, output $\boldsymbol{\sigma}^{\mathsf{enc}} = \mathbf{B} \cdot \mathbf{r} + g(\mathbf{d})$ NIZK proof $\pi$

**Verify$_{\mathsf{vk}}$(msg, $\boldsymbol{\sigma}$)**

Apply $f^i_{\mathsf{vk,msg}}$ homomorphically on $\boldsymbol{\sigma}^{\mathsf{enc}}$

Get $\boldsymbol{\sigma}_i^{\mathsf{enc}} = \mathbf{B} \cdot \mathbf{r}_i + g\left(f^i_{\mathsf{vk,msg}}(\mathbf{d})\right)$

Check that $f^i_{\mathsf{vk,msg}}(\mathbf{d}) \in S_i$ and that NIZK proof $\pi$ is valid

**Regev SPS Encryption**

Public-key matrix $\mathbf{B} \in \mathbb{Z}_q^{d \times \tau}$

invertible additive homomorphic encoding $g : \mathscr{M} \to \mathbb{Z}_q^d$

Sample randomness $\mathbf{r} \leftarrow_r \mathscr{R}$

Ciphertext will be $\mathsf{ct} = \mathbf{B} \cdot \mathbf{r} + g(\mathsf{msg})$

**Verify$_{\mathsf{vk}}$(msg, $\boldsymbol{\sigma}$) for Boyen SPS**

Let $f^1_{\mathsf{vk,msg}}(\boldsymbol{\sigma}) = \mathbf{F}_{\mathsf{msg}} \cdot \boldsymbol{\sigma}$, $f^2_{\mathsf{vk,msg}}(\boldsymbol{\sigma}) = \boldsymbol{\sigma}$

Consider structure-preserving sets $S_1 = \{\mathbf{0}\}$

and $S_2$, a ball of short vectors.

Output 1 if $f^i_{\mathsf{vk,msg}}(\boldsymbol{\sigma}) = S_i$ for both $i = \{1,2\}$

computable since $g$ is homomorphic

where $S$ is structure preserving

We need a NIZK to check that a ct is of the form Enc(msg) where msg $\in S$

Linea

# Structure-Preserving NIZK

We need a NIZK to check that a $\mathsf{ct}$ is of the form $\mathsf{Enc}(\mathsf{msg})$ where $\mathsf{msg} \in S$.

We adapt the sigma protocol of [Libert et al. 2020]

# From Structure-Preserving $\Sigma$-Protocol to NIZK

Option 1: Use Fiat-Shamir

Option 2: use correlation-intractable hashing to obtain security in the standard model.

[Libert et al. 2020] uses CI-Hashing for $\mathsf{NC}_1$ circuits

# Recap: Encryption of a Signature

**KeyGen**

Generate matrix $\mathbf{A}$ and short trapdoor $\mathbf{T}_{\mathbf{A}}$

Sample uniform $(\mathbf{C}_0 \ldots \mathbf{C}_\ell)$

Output keys vk $= (\mathbf{A}, \mathbf{C}_0 \ldots \mathbf{C}_\ell)$, sk $= \mathbf{T}_{\mathbf{A}}$

**$\text{Sign}_{\text{sk}}(\text{msg})$**

Compute $\mathbf{C}_{\text{msg}} = \mathbf{C}_0 + \sum_{i=1}^{\ell} \text{msg}_i \mathbf{C}_i$

Set $\mathbf{F}_{\mathbf{msg}} = [\mathbf{A} \,|\, \mathbf{C}_{\text{msg}}]$

Use $\mathbf{T}_{\mathbf{A}}$ to generate short $\boldsymbol{\sigma}$ such that $\mathbf{F}_{\text{msg}} \cdot \boldsymbol{\sigma} = \mathbf{0}$

Sample $\mathbf{r}$, output $\boldsymbol{\sigma}^{\text{enc}} = \mathbf{B}_\alpha \cdot \mathbf{r} + g_\alpha(\mathbf{d})$ NIZK proof $\pi$

**$\text{Verify}_{\text{vk}}(\text{msg}, \boldsymbol{\sigma})$**

Apply $f^i_{\text{vk,msg}}$ homomorphically on $\boldsymbol{\sigma}^{\text{enc}}$

Get $\boldsymbol{\sigma}_i^{\text{enc}} = \mathbf{B} \cdot \mathbf{r}_i + g\left(f^i_{\text{vk,msg}}(\mathbf{d})\right)$

Check that $f^i_{\text{vk,msg}}(\mathbf{d}) \in S_i$ and that NIZK proof $\pi$ is valid

**Regev SPS Encryption**

Public-key matrix $\mathbf{B} \in \mathbb{Z}_q^{d \times \tau}$

invertible additive homomorphic encoding $g : \mathscr{M} \to \mathbb{Z}_q^d$

Sample randomness $\mathbf{r} \leftarrow_r \mathscr{R}$

Ciphertext will be ct $= \mathbf{B} \cdot \mathbf{r} + g(\text{msg})$

**$\text{Verify}_{\text{vk}}(\text{msg}, \boldsymbol{\sigma})$ for Boyen SPS**

Let $f^1_{\text{vk,msg}}(\boldsymbol{\sigma}) = \mathbf{F}_{\text{msg}} \cdot \boldsymbol{\sigma}$, $f^2_{\text{vk,msg}}(\boldsymbol{\sigma}) = \boldsymbol{\sigma}$

Consider structure-preserving sets $S_1 = \{\mathbf{0}\}$

and $S_2$, a ball of short vectors.

Output 1 if $f^i_{\text{vk,msg}}(\boldsymbol{\sigma}) = S_i$ for both $i = \{1,2\}$

computable since $g_\alpha$ is homomorphic

where $S$ is structure preserving

We now have a NIZK to check that a ct is of the form Enc(msg) where msg $\in S$

Linea

# Structure-Preserving Cryptography (SPS)

In structure-preserving cryptography, our NIZK efficiently proves statements of the form:

- a ciphertext encrypts a valid signature. ← we have shown how to do this
- a signature signs a valid ciphertext. ← a NIZK proof certifies that the ciphertext is valid

# Application: Verifiable Encrypted Signature (VES)

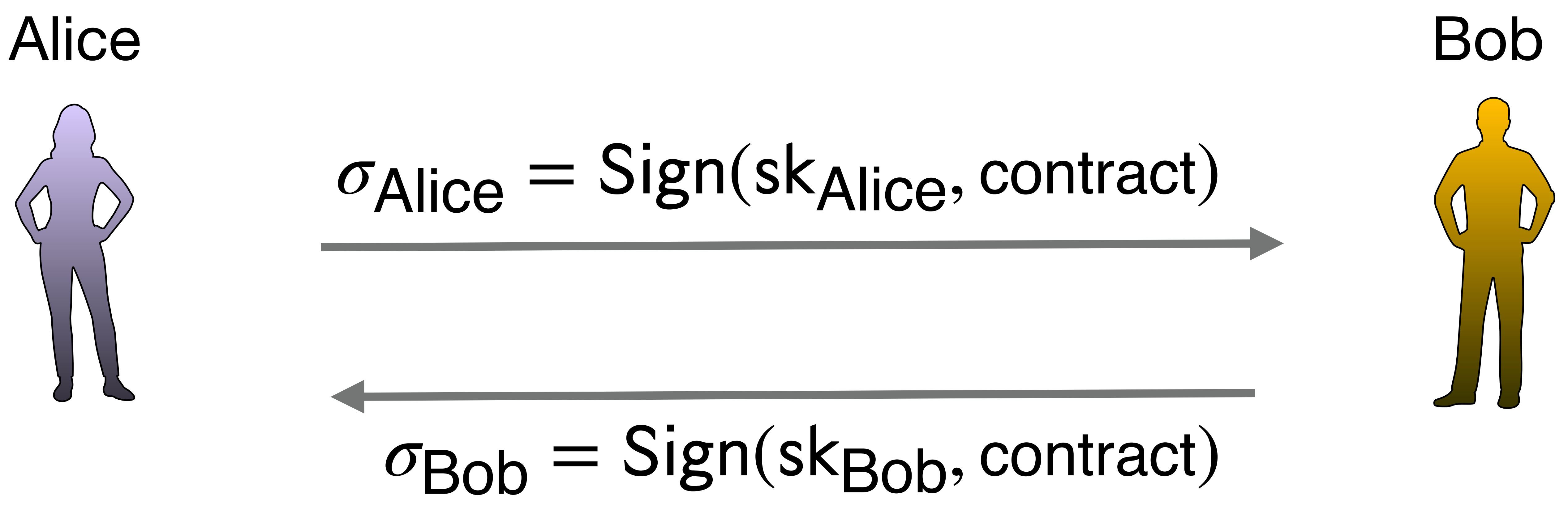
Prove that an encrypted signature is valid without revealing the signature.

Our construction is the most efficient lattice-based VES in the standard model.

Linea

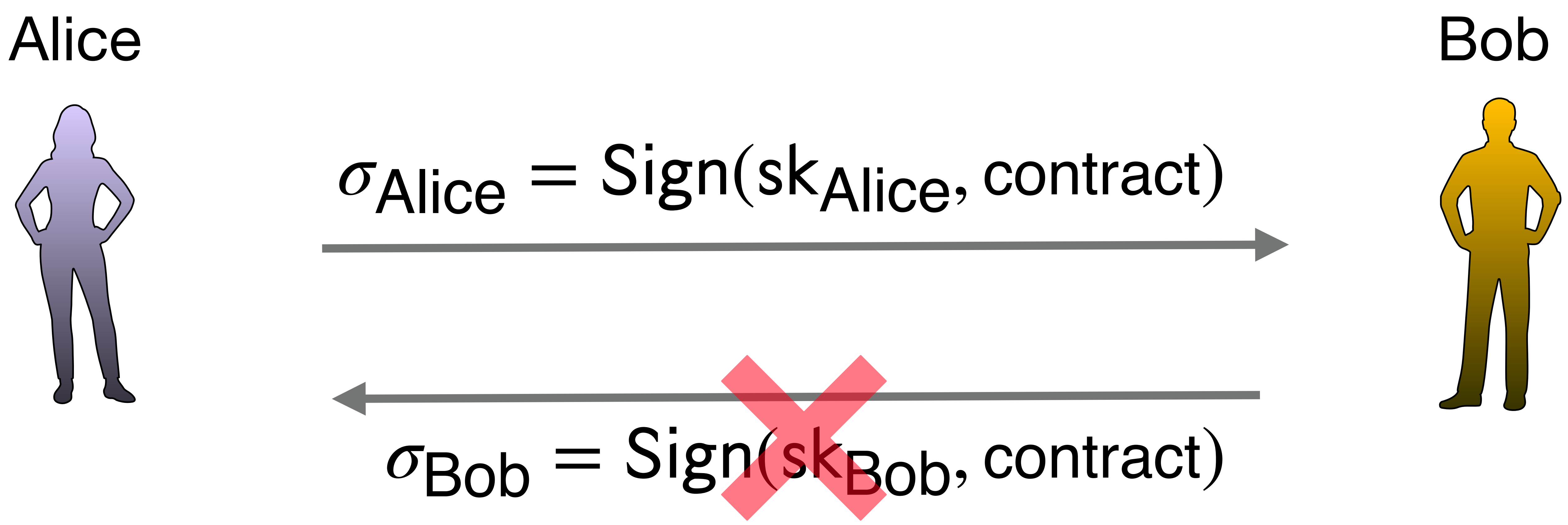
# Motivation: Contract Signing

Alice and Bob want to sign a contract.



# Motivation: Contract Signing

Alice and Bob want to sign a contract.

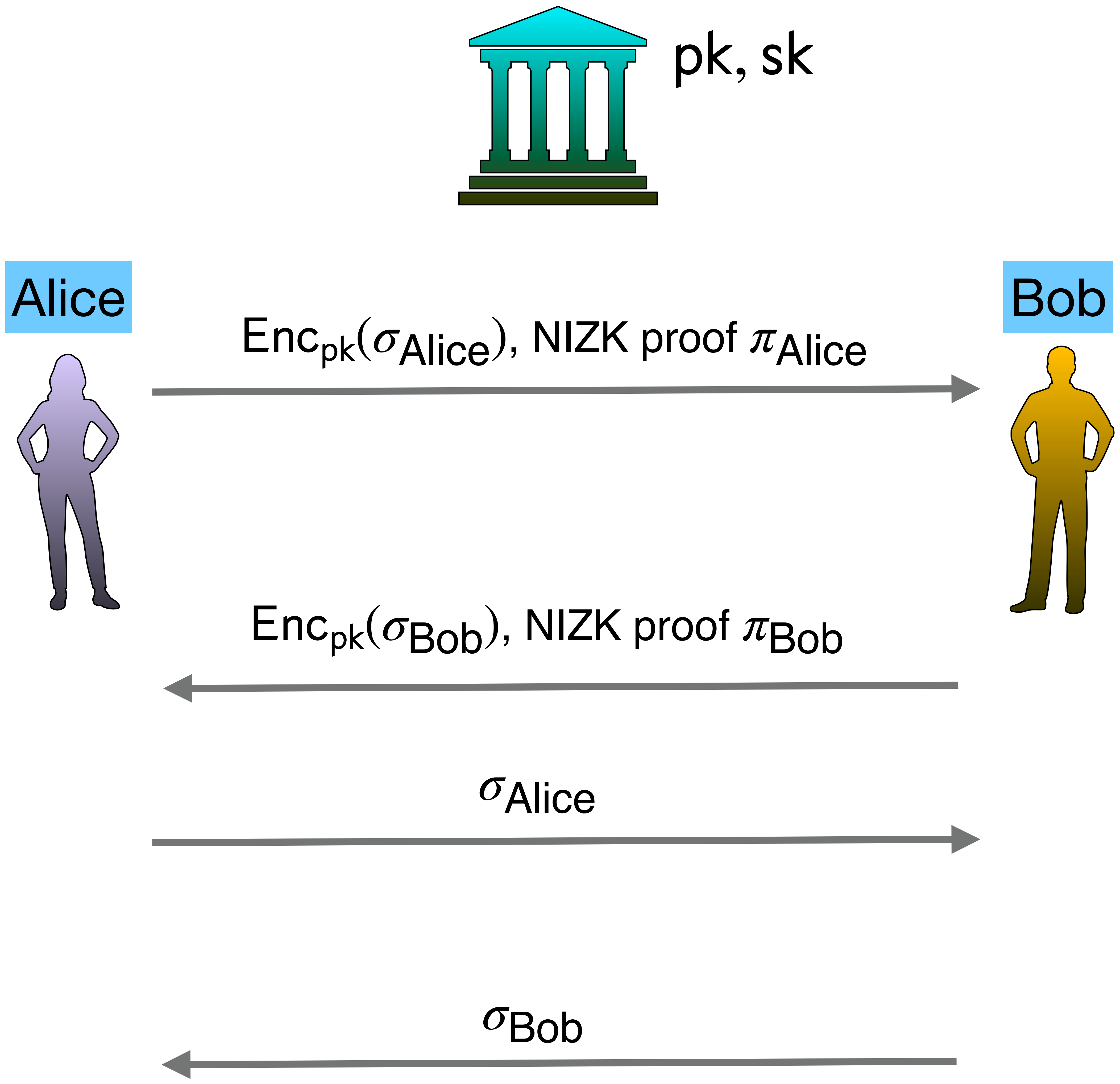



Bob refuses to sign and instead forwards $\sigma_{\text{Alice}}$ to a third party to negotiate.

Bob can impersonate Alice to another third party.

# Motivation: Contract Signing

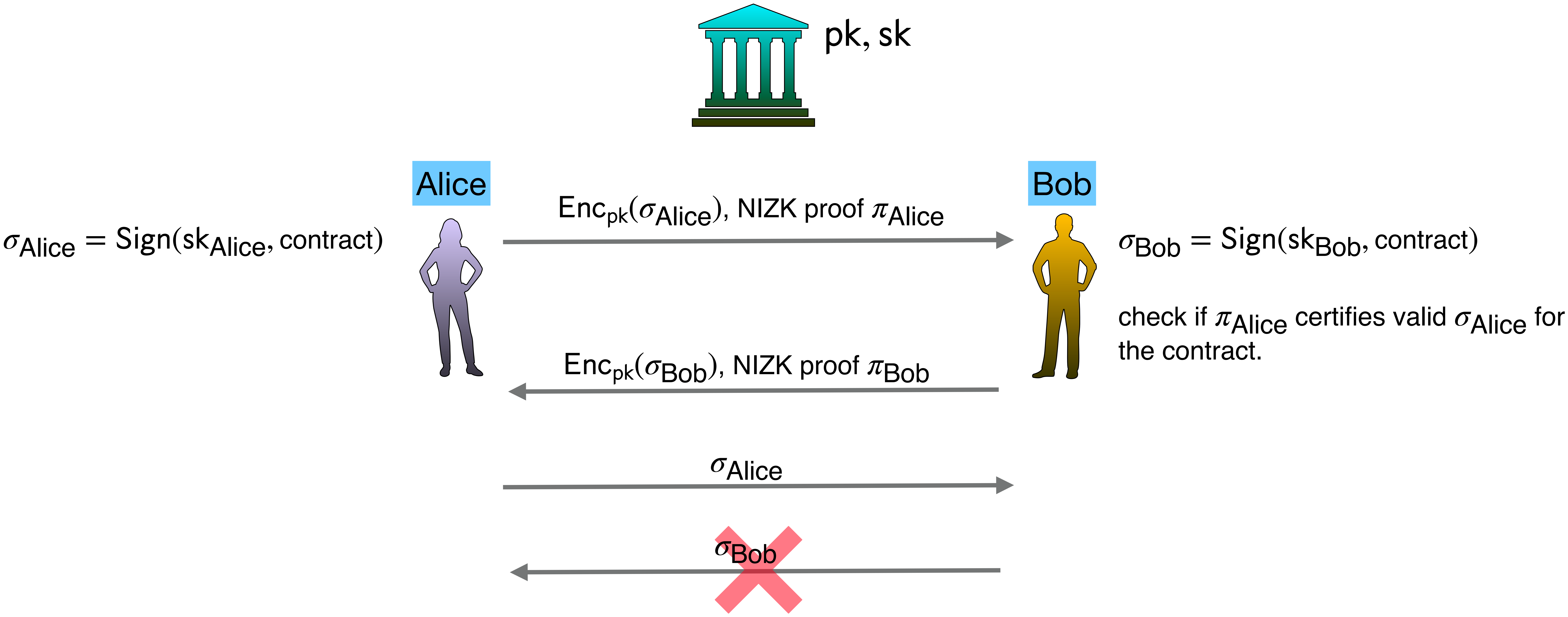
$\sigma_{\text{Alice}} = \text{Sign}(\text{sk}_{\text{Alice}}, \text{contract})$




$\sigma_{\text{Bob}} = \text{Sign}(\text{sk}_{\text{Bob}}, \text{contract})$

check if $\pi_{\text{Alice}}$ certifies valid $\sigma_{\text{Alice}}$ for the contract.

# Motivation: Contract Signing




If any party refuses to sign, the other party forwards the encryption and proof to the authority which will decrypt.

# Contributions

We put forward:

- a unifying framework for structure-preserving cryptography for lattices.

| Signatures | Encryption |
| --- | --- |
| [Boyen10] | Regev Encryption [Regev05] |
| [Rúckert10] | Dual Regev [GPV08] |
| new modification of [Rúckert10] using [ABB10] delegation | [GSW13] |

# Contributions

We put forward:

- a unifying framework for structure-preserving cryptography for lattices.

| Signatures | Encryption |
|---|---|
| [Boyen10] | Regev Encryption [Regev05] |
| [Rúckert10] | Dual Regev [GPV08] |
| new modification of [Rúckert10] using [ABB10] delegation | [GSW13] |

- structure-preserving NIZK, generalising a protocol from [Libert et al. 2020]

Linea

# Contributions

We put forward: • a unifying framework for structure-preserving cryptography for lattices.

| Signatures | Encryption |
|---|---|
| [Boyen10] | Regev Encryption [Regev05] |
| [Rúckert10] | Dual Regev [GPV08] |
| new modification of [Rúckert10] using [ABB10] delegation | [GSW13] |

- structure-preserving NIZK, generalising a protocol from [Libert et al. 2020]

- application to verifiable encrypted signature (VES).
  - new proof, similar to [Fuchsbauer2011] with several new technical details.
  - currently the most efficient lattice VES in the standard model

# Limitation and an Open Problem

| | new ISIS-based signature | [Rückert10]'s signature | [Boyen10] |
|---|---|---|---|
| [Regev05] | compatible | compatible | incompatible |
| [GPV08] | compatible | compatible | incompatible |
| [GSW13] | compatible | compatible | compatible |

# Thank you for your attention!

## Questions?