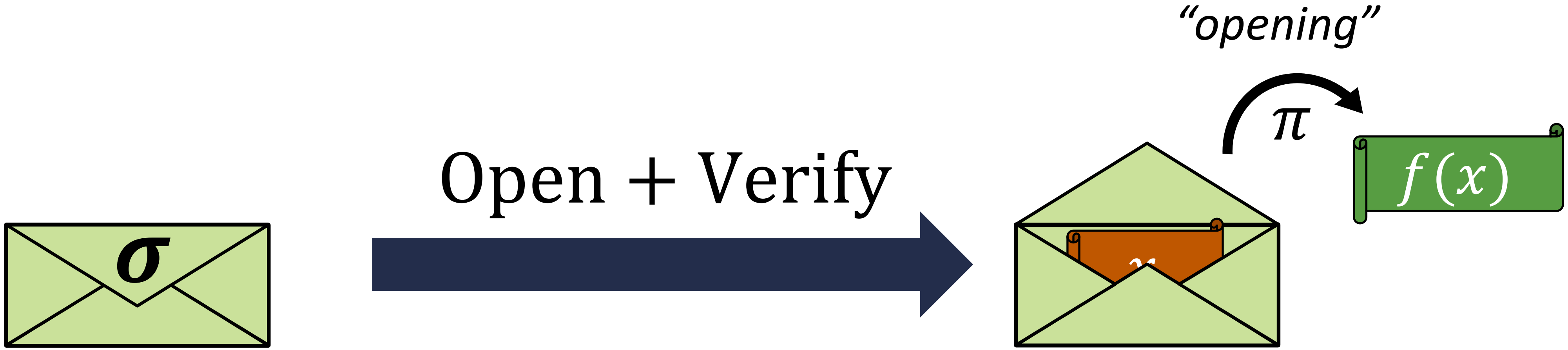
Lattice-Based Functional Commitments: Constructions and Cryptanalysis

David Wu

December 2023

based on joint work with Hoeteck Wee

Functional Commitments



Functional Commitments

$$\mathrm{Commit}(\mathrm{crs}, x) \to (\sigma, \mathrm{st})$$

Takes a common reference string and commits to an input x

Outputs commitment σ and commitment state st

Functional Commitments

$$\mathrm{Commit}(\mathrm{crs}, x) \rightarrow (\sigma, \mathrm{st})$$

$$\mathrm{Open}(\mathrm{st}, f) \rightarrow \pi$$

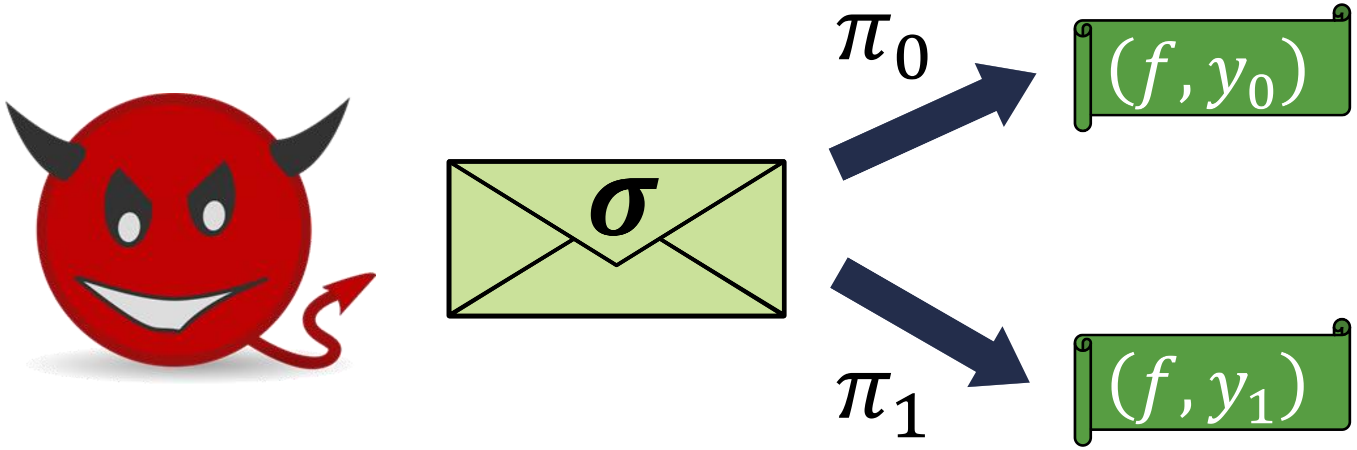
Takes the commitment state and a function f and outputs an opening π

$$\mathrm{Verify}(\mathrm{crs}, \sigma, (f, y), \pi) \rightarrow 0/1$$

Checks whether π is valid opening of σ to value y with respect to f

Functional Commitments

Binding: efficient adversary cannot open σ to two different values with respect to the **same** f



$\text{Verify}(\text{crs}, \sigma, (f, y_0), \pi_0) = 1$

$\text{Verify}(\text{crs}, \sigma, (f, y_1), \pi_1) = 1$

Functional Commitments

Succinctness: commitments and openings should be short

- **Short commitment:** $|\sigma| = \text{poly}(\lambda, \log|x|)$
- **Short opening:** $|\pi| = \text{poly}(\lambda, \log|x|, |f(x)|)$

Will consider relaxation where $|\sigma|$ and $|\pi|$ can grow with **depth** of the circuit computing f

Special Cases of Functional Commitments

Vector commitments:

$[x_1, x_2, \ldots, x_n]$ $\xrightarrow{\mathrm{ind}_i(x_1, \ldots, x_n) \coloneqq x_i}$ x_i

commit to a vector, open at an index

Polynomial commitments:

$[\alpha_0, \alpha_1, \ldots, \alpha_d]$ $\xrightarrow{f_x(\alpha_0, \ldots, \alpha_d) \coloneqq \alpha_0 + \alpha_1 x + \cdots + \alpha_d x^d}$ y

commit to a polynomial, open to the evaluation at x

Succinct Functional Commitments

(not an exhaustive list!)

Scheme	Function Class	Assumption
[Mer87]	vector commitment	collision-resistant hash functions
[LY10, CF13, LM19, GRWZ20]	vector commitment	q-type pairing assumptions
[CF13, LM19, BBF19]	vector commitment	groups of unknown order
[PPS21]	vector commitment	short integer solutions (SIS)
[KZG10, Lee20]	polynomial commitment	q-type pairing assumptions
[BFS19, BHRRS21, BF23]	polynomial commitment	groups of unknown order
[LRY16]	linear functions	q-type pairing assumptions
[ACLMT22]	constant-degree polynomials	k-R-ISIS assumption (falsifiable)
[LRY16]	Boolean circuits	collision-resistant hash functions + SNARKs
[dCP23]	Boolean circuits	SIS (non-succinct openings in general)
[KLVW23]	Boolean circuits	LWE (via batch arguments)
[BCFL23]	Boolean circuits	twin k-R-ISIS
[WW23a, WW23b]	Boolean circuits	ℓ-succinct SIS **This talk**

Framework for Lattice Commitments

Captures and generalizes other lattice-based functional commitments [PPS21, ACLMT22]

Common reference string (for inputs of length ℓ):

matrices $\boldsymbol{A}_1, \dots, \boldsymbol{A}_\ell \in \mathbb{Z}_q^{n \times m}$

target vectors $\boldsymbol{t}_1, \dots, \boldsymbol{t}_\ell \in \mathbb{Z}_q^n$

auxiliary data: cross-terms $\boldsymbol{u}_{ij} \leftarrow \boldsymbol{A}_i^{-1}(\boldsymbol{t}_j) \in \mathbb{Z}_q^m$ where $i \neq j$

short (i.e., low-norm) vector satisfying $\boldsymbol{A}_i \boldsymbol{u}_{ij} = \boldsymbol{t}_j$

Framework for Lattice Commitments

Captures and generalizes other lattice-based functional commitments [PPS21, ACLMT22]

Common reference string (for inputs of length ℓ):

matrices $\boldsymbol{A}_1, \ldots, \boldsymbol{A}_\ell \in \mathbb{Z}_q^{n \times m}$

target vectors $\boldsymbol{t}_1, \ldots, \boldsymbol{t}_\ell \in \mathbb{Z}_q^n$

auxiliary data: cross-terms $\boldsymbol{u}_{ij} \leftarrow \boldsymbol{A}_i^{-1}(\boldsymbol{t}_j) \in \mathbb{Z}_q^m$ where $i \neq j$

Commitment to $\boldsymbol{x} \in \mathbb{Z}_q^\ell$:

$$\boldsymbol{c} = \sum_{i \in [\ell]} x_i \boldsymbol{t}_i$$

linear combination of target vectors

Opening to value y at index i:

short $\boldsymbol{v}_i$ such that $\boldsymbol{c} = \boldsymbol{A}_i \boldsymbol{v}_i + y \cdot \boldsymbol{t}_i$

Honest opening:

$$\boldsymbol{v}_i = \sum_{j \neq i} x_j \boldsymbol{u}_{ij}$$

Correct as long as $\boldsymbol{x}$ is short

$$\boldsymbol{A}_i \boldsymbol{v}_i + x_i \boldsymbol{t}_i = \sum_{j \neq i} x_j \boldsymbol{A}_i \boldsymbol{u}_{ij} + x_i \boldsymbol{t}_i = \sum_{j \in [\ell]} x_j \boldsymbol{t}_j = \boldsymbol{c}$$

Framework for Lattice Commitments

Captures and generalizes other lattice-based functional commitments [PPS21, ACLMT22]

Common reference string (for inputs of length ℓ):

matrices $\boldsymbol{A}_1, \dots, \boldsymbol{A}_\ell \in \mathbb{Z}_q^{n \times m}$

target vectors $\boldsymbol{t}_1, \dots, \boldsymbol{t}_\ell \in \mathbb{Z}_q^n$

auxiliary data: cross-terms $\boldsymbol{u}_{ij} \leftarrow \boldsymbol{A}_i^{-1}(\boldsymbol{t}_j) \in \mathbb{Z}_q^m$ where $i \neq j$

[PPS21]: $\boldsymbol{A}_i \leftarrow \mathbb{Z}_q^{n \times m}$ and $\boldsymbol{t}_i \leftarrow \mathbb{Z}_q^n$ are independent and uniform

suffices for vector commitments (from SIS)

[ACLMT21]: $\boldsymbol{A}_i = \boldsymbol{W}_i \boldsymbol{A}$ and $\boldsymbol{t}_i = \boldsymbol{W}_i \boldsymbol{u}_i$ where $\boldsymbol{W}_i \leftarrow \mathbb{Z}_q^{n \times n}, \boldsymbol{A} \leftarrow \mathbb{Z}_q^{n \times m}, \boldsymbol{u}_i \leftarrow \mathbb{Z}_q^n$

(one candidate adaptation to the integer case)

<u>generalizes</u> to functional commitments for constant-degree polynomials (from k-R-ISIS)

Our Approach

Captures and generalizes other lattice-based functional commitments [PPS21, ACLMT22]

Verification invariant: $\boldsymbol{c} = \boldsymbol{A}_i \boldsymbol{v}_i + x_i \boldsymbol{t}_i \qquad \forall i \in [\ell]$

for a short $\boldsymbol{v}_i$

Our approach: rewrite ℓ relations as a single linear system

$$\begin{bmatrix} \boldsymbol{A}_1 & & & -\boldsymbol{I}_n \\ & \ddots & & \vdots \\ & & \boldsymbol{A}_\ell & -\boldsymbol{I}_n \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{v}_1 \\ \vdots \\ \boldsymbol{v}_\ell \\ \boldsymbol{c} \end{bmatrix} = \begin{bmatrix} -x_1 \boldsymbol{t}_1 \\ \vdots \\ -x_\ell \boldsymbol{t}_\ell \end{bmatrix}$$

$\boldsymbol{I}_n$ denotes the identity matrix

Our Approach

Captures and generalizes other lattice-based functional commitments [PPS21, ACLMT22]

Verification invariant: $\boldsymbol{c} = \boldsymbol{A}_i \boldsymbol{v}_i + x_i \boldsymbol{t}_i \qquad \forall i \in [\ell]$

for a short $\boldsymbol{v}_i$

Our approach: rewrite ℓ relations as a single linear system

$$\left[\begin{array}{ccc|c} \boldsymbol{A}_1 & & & -\boldsymbol{G} \\ & \ddots & & \vdots \\ & & \boldsymbol{A}_\ell & -\boldsymbol{G} \end{array}\right] \cdot \begin{bmatrix} \boldsymbol{v}_1 \\ \vdots \\ \boldsymbol{v}_\ell \\ \hat{\boldsymbol{c}} \end{bmatrix} = \begin{bmatrix} -x_1 \boldsymbol{t}_1 \\ \vdots \\ -x_\ell \boldsymbol{t}_\ell \end{bmatrix}$$

"powers of two matrix"

$$\boldsymbol{G} = \begin{bmatrix} 1 & 2 & \cdots & 2^{\lfloor \log q \rfloor} & & & & \\ & & & & \ddots & & & \\ & & & & 1 & 2 & \cdots & 2^{\lfloor \log q \rfloor} \end{bmatrix}$$

For security and functionality, it will be useful to write $\boldsymbol{c} = \boldsymbol{G}\hat{\boldsymbol{c}}$

Our Approach

Captures and generalizes other lattice-based functional commitments [PPS21, ACLMT22]

Verification invariant: $\boldsymbol{c} = \boldsymbol{A}_i \boldsymbol{v}_i + x_i \boldsymbol{t}_i \qquad \forall i \in [\ell]$

for a short $\boldsymbol{v}_i$

Our approach: rewrite ℓ relations as a single linear system

$$\left[\begin{array}{ccc|c} \boldsymbol{A}_1 & & & -\boldsymbol{G} \\ & \ddots & & \vdots \\ & & \boldsymbol{A}_\ell & -\boldsymbol{G} \end{array}\right] \cdot \begin{bmatrix} \boldsymbol{v}_1 \\ \vdots \\ \boldsymbol{v}_\ell \\ \hat{\boldsymbol{c}} \end{bmatrix} = \begin{bmatrix} -x_1 \boldsymbol{t}_1 \\ \vdots \\ -x_\ell \boldsymbol{t}_\ell \end{bmatrix}$$

Common reference string:

- matrices $\boldsymbol{A}_1, \dots, \boldsymbol{A}_\ell \in \mathbb{Z}_q^{n \times m}$
- target vectors $\boldsymbol{t}_1, \dots, \boldsymbol{t}_\ell \in \mathbb{Z}_q^n$
- *auxiliary data:* cross-terms $\boldsymbol{u}_{ij} \leftarrow \boldsymbol{A}_i^{-1}(\boldsymbol{t}_j)$

Our Approach

Captures and generalizes other lattice-based functional commitments [PPS21, ACLMT22]

Verification invariant: $\boldsymbol{c} = \boldsymbol{A}_i \boldsymbol{v}_i + x_i \boldsymbol{t}_i \qquad \forall i \in [\ell]$

for a short $\boldsymbol{v}_i$

Our approach: rewrite ℓ relations as a single linear system

$$\underbrace{\left[\begin{array}{ccc|c} \boldsymbol{A}_1 & & & -\boldsymbol{G} \\ & \ddots & & \vdots \\ & & \boldsymbol{A}_\ell & -\boldsymbol{G} \end{array}\right]}_{\boldsymbol{B}_\ell} \cdot \begin{bmatrix} \boldsymbol{v}_1 \\ \vdots \\ \boldsymbol{v}_\ell \\ \hat{\boldsymbol{c}} \end{bmatrix} = \begin{bmatrix} -x_1 \boldsymbol{t}_1 \\ \vdots \\ -x_\ell \boldsymbol{t}_\ell \end{bmatrix}$$

Common reference string:

- matrices $\boldsymbol{A}_1, \dots, \boldsymbol{A}_\ell \in \mathbb{Z}_q^{n \times m}$
- target vectors $\boldsymbol{t}_1, \dots, \boldsymbol{t}_\ell \in \mathbb{Z}_q^n$
- *auxiliary data:* ~~cross-terms $\boldsymbol{u}_{ij} \leftarrow \boldsymbol{A}_i^{-1}(\boldsymbol{t}_j)$~~ trapdoor for $\boldsymbol{B}_\ell$

Trapdoor for $\boldsymbol{B}_\ell$ can be used to sample short solutions $\boldsymbol{x}$ to the linear system $\boldsymbol{B}_\ell \boldsymbol{x} = \boldsymbol{y}$ (for arbitrary $\boldsymbol{y}$)

Our Approach

Captures and generalizes other lattice-based functional commitments [PPS21, ACLMT22]

Verification invariant: $\boldsymbol{c} = \boldsymbol{A}_i \boldsymbol{v}_i + x_i \boldsymbol{t}_i \qquad \forall i \in [\ell]$

for a short $\boldsymbol{v}_i$

Our approach: rewrite ℓ relations as a single linear system

$$\underbrace{\begin{bmatrix} \boldsymbol{A}_1 & & & -\boldsymbol{G} \\ & \ddots & & \vdots \\ & & \boldsymbol{A}_\ell & -\boldsymbol{G} \end{bmatrix}}_{\boldsymbol{B}_\ell} \cdot \begin{bmatrix} \boldsymbol{v}_1 \\ \vdots \\ \boldsymbol{v}_\ell \\ \hat{\boldsymbol{c}} \end{bmatrix} = \begin{bmatrix} -x_1 \boldsymbol{t}_1 \\ \vdots \\ -x_\ell \boldsymbol{t}_\ell \end{bmatrix}$$

Committing to an input $\boldsymbol{x}$:

Use trapdoor for $\boldsymbol{B}_\ell$ to **jointly** sample a solution $\boldsymbol{v}_1, \dots, \boldsymbol{v}_\ell, \hat{\boldsymbol{c}}$

$\boldsymbol{c} = \boldsymbol{G}\hat{\boldsymbol{c}}$ is the commitment and $\boldsymbol{v}_1, \dots \boldsymbol{v}_\ell$ are the openings

Supports commitments to arbitrary (i.e., large) values over $\mathbb{Z}_q$

Our Approach

Captures and generalizes other lattice-based functional commitments [PPS21, ACLMT22]

Verification invariant: $\boldsymbol{c} = \boldsymbol{A}_i \boldsymbol{v}_i + x_i \boldsymbol{t}_i \qquad \forall i \in [\ell]$

for a short $\boldsymbol{v}_i$

Our approach: rewrite ℓ relations as a single linear system

$$\underbrace{\left[\begin{array}{ccc|c} \boldsymbol{A}_1 & & & -\boldsymbol{G} \\ & \ddots & & \vdots \\ & & \boldsymbol{A}_\ell & -\boldsymbol{G} \end{array}\right]}_{\boldsymbol{B}_\ell} \cdot \begin{bmatrix} \boldsymbol{v}_1 \\ \vdots \\ \boldsymbol{v}_\ell \\ \hat{\boldsymbol{c}} \end{bmatrix} = \begin{bmatrix} -x_1 \boldsymbol{t}_1 \\ \vdots \\ -x_\ell \boldsymbol{t}_\ell \end{bmatrix}$$

Committing to an input $\boldsymbol{x}$:

Use trapdoor for $\boldsymbol{B}_\ell$ to **jointly** sample a solution $\boldsymbol{v}_1, \dots, \boldsymbol{v}_\ell, \hat{\boldsymbol{c}}$

$\boldsymbol{c} = \boldsymbol{G}\hat{\boldsymbol{c}}$ is the commitment and $\boldsymbol{v}_1, \dots \boldsymbol{v}_\ell$ are the openings

Supports statistically private openings
(commitment + opening *hides* unopened positions)

Proving Security

Captures and generalizes other lattice-based functional commitments [PPS21, ACLMT22]

Verification invariant: $\boldsymbol{c} = \boldsymbol{A}_i \boldsymbol{v}_i + x_i \boldsymbol{t}_i \qquad \forall i \in [\ell]$

for a short $\boldsymbol{v}_i$

Suppose adversary can break binding

outputs $\boldsymbol{c}, (\boldsymbol{v}_i, x_i), (\boldsymbol{v}_i', x_i')$ such that

$$\boldsymbol{c} = \boldsymbol{A}_i \boldsymbol{v}_i + x_i \boldsymbol{t}_i$$
$$= \boldsymbol{A}_i \boldsymbol{v}_i' + x_i' \boldsymbol{t}_i$$

given matrices $\boldsymbol{A}_1, \dots, \boldsymbol{A}_\ell$

target vectors $\boldsymbol{t}_1, \dots, \boldsymbol{t}_\ell$

trapdoor for $\boldsymbol{B}_\ell$

set $\boldsymbol{A}_i \leftarrow \mathbb{Z}_q^{n \times m}$

set $\boldsymbol{t}_i = \boldsymbol{e}_1 = [1,0, \dots, 0]^{\mathrm{T}}$

Short integer solutions (SIS)

given $\boldsymbol{A} \leftarrow \mathbb{Z}_q^{n \times m}$, hard to find short $\boldsymbol{x} \neq 0$ such that $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}$

$$\boldsymbol{A}_i (\boldsymbol{v}_i - \boldsymbol{v}_i') = (x_i - x_i') \boldsymbol{e}_1$$

$\boldsymbol{v}_i - \boldsymbol{v}_i'$ is a SIS solution for $\boldsymbol{A}_i$ without the first row

Proving Security

Captures and generalizes other lattice-based functional commitments [PPS21, ACLMT22]

Verification invariant: $\boldsymbol{c} = \boldsymbol{A}_i \boldsymbol{v}_i + x_i \boldsymbol{t}_i \qquad \forall i \in [\ell]$

for a short $\boldsymbol{v}_i$

Adversary that breaks binding can solve SIS with respect to $\boldsymbol{A}_i$

(technically $\boldsymbol{A}_i$ *without the first row – which is equivalent to SIS with dimension* $n - 1$*)*

but… adversary also gets additional information beyond $\boldsymbol{A}_i$

$$\boldsymbol{B}_\ell = \left[\begin{array}{ccc|c} \boldsymbol{A}_1 & & & -\boldsymbol{G} \\ & \ddots & & \vdots \\ & & \boldsymbol{A}_\ell & -\boldsymbol{G} \end{array}\right]$$

Adversary sees **trapdoor** for $\boldsymbol{B}_\ell$

Basis-Augmented SIS (BASIS) Assumption

Captures and generalizes other lattice-based functional commitments [PPS21, ACLMT22]

Verification invariant: $\boldsymbol{c} = \boldsymbol{A}_i \boldsymbol{v}_i + x_i \boldsymbol{t}_i \qquad \forall i \in [\ell]$

for a short $\boldsymbol{v}_i$

Adversary that breaks binding can solve SIS with respect to $\boldsymbol{A}_i$

Basis-augmented SIS (BASIS) assumption:

SIS is hard with respect to $\boldsymbol{A}_i$

given a trapdoor (a basis) for the matrix

$$\boldsymbol{B}_\ell = \left[\begin{array}{ccc|c} \boldsymbol{A}_1 & & & -\boldsymbol{G} \\ & \ddots & & \vdots \\ & & \boldsymbol{A}_\ell & -\boldsymbol{G} \end{array}\right]$$

Can simulate CRS from BASIS challenge:

matrices $\boldsymbol{A}_1, \ldots, \boldsymbol{A}_\ell \leftarrow \mathbb{Z}_q^{n \times m}$

trapdoor for $\boldsymbol{B}_\ell$

Basis-Augmented SIS (BASIS) Assumption

SIS is hard with respect to $\boldsymbol{A}_i$ given a trapdoor (a basis) for the matrix

$$\boldsymbol{B}_\ell = \left[\begin{array}{ccc|c} \boldsymbol{A}_1 & & & -\boldsymbol{G} \\ & \ddots & & \vdots \\ & & \boldsymbol{A}_\ell & -\boldsymbol{G} \end{array}\right]$$

When $\boldsymbol{A}_1, \dots, \boldsymbol{A}_\ell \leftarrow \mathbb{Z}_q^{n \times m}$ are uniform and independent:

hardness of SIS implies hardness of BASIS

(follows from standard lattice trapdoor extension techniques)

Vector Commitments from SIS

Common reference string (for inputs of length ℓ):

matrices $\boldsymbol{A}_1, \ldots, \boldsymbol{A}_\ell \in \mathbb{Z}_q^{n \times m}$

auxiliary data: trapdoor for $\boldsymbol{B}_\ell = \begin{bmatrix} \boldsymbol{A}_1 & & & -\boldsymbol{G} \\ & \ddots & & \vdots \\ & & \boldsymbol{A}_\ell & -\boldsymbol{G} \end{bmatrix}$

To commit to a vector $\boldsymbol{x} \in \mathbb{Z}_q^\ell$: sample solution $(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_\ell, \hat{\boldsymbol{c}})$

$$\begin{bmatrix} \boldsymbol{A}_1 & & & -\boldsymbol{G} \\ & \ddots & & \vdots \\ & & \boldsymbol{A}_\ell & -\boldsymbol{G} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{v}_1 \\ \vdots \\ \boldsymbol{v}_\ell \\ \hat{\boldsymbol{c}} \end{bmatrix} = \begin{bmatrix} -x_1 \boldsymbol{e}_1 \\ \vdots \\ -x_\ell \boldsymbol{e}_\ell \end{bmatrix}$$

Commitment is $\boldsymbol{c} = \boldsymbol{G}\hat{\boldsymbol{c}}$ Openings are $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_\ell$

Can commit and open to **arbitrary** $\mathbb{Z}_q$ vectors

Commitments and openings statistically **hide** unopened components

Linearly homomorphic:
$\boldsymbol{c} + \boldsymbol{c}'$ is a commitment to $\boldsymbol{x} + \boldsymbol{x}'$ with openings $\boldsymbol{v}_i + \boldsymbol{v}_i'$

Functional Commitments for Circuits

Setting: commit to an input $\boldsymbol{x} \in \{0,1\}^\ell$, open to $f(\boldsymbol{x})$

(f can be an arbitrary Boolean circuit)

Will need some basic lattice machinery for homomorphic computation

[GSW13, BGGHNSVV14, GVW15]

Let $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$ be an arbitrary matrix

$$\boldsymbol{C}_1 = \boldsymbol{A}\boldsymbol{V}_1 + x_1 \boldsymbol{G}$$

$$\vdots$$

$$\boldsymbol{C}_\ell = \boldsymbol{A}\boldsymbol{V}_\ell + x_\ell \boldsymbol{G}$$

homomorphic evaluation →

$$\boldsymbol{C}_f = \boldsymbol{A}\boldsymbol{V}_f + f(\boldsymbol{x}) \cdot \boldsymbol{G}$$

$\boldsymbol{C}_i$ is an encoding of x_i with (short) randomness $\boldsymbol{V}_i$

$\boldsymbol{C}_f$ is an encoding of $f(\boldsymbol{x})$ with (short) randomness $\boldsymbol{V}_f$

Functional Commitments for Circuits

Replace random $\boldsymbol{A}_i$ with a single $\boldsymbol{A}$ (and gadget matrix with $\boldsymbol{W}_1, \dots, \boldsymbol{W}_\ell$)

$$\boldsymbol{A} \leftarrow \mathbb{Z}_q^{n \times m} \quad , \quad \boldsymbol{A}_i \coloneqq \boldsymbol{A}$$

$$\boldsymbol{W}_1, \dots, \boldsymbol{W}_\ell \leftarrow \mathbb{Z}_q^{n \times n}$$

Common reference string contains trapdoor for matrix $\boldsymbol{B}_\ell$:

$$\boldsymbol{B}_\ell = \left[\begin{array}{ccc|c} \boldsymbol{A} & & & \boldsymbol{W}_1 \\ & \ddots & & \vdots \\ & & \boldsymbol{A} & \boldsymbol{W}_\ell \end{array}\right]$$

Functional Commitments for Circuits

Replace random $\boldsymbol{A}_i$ with a single $\boldsymbol{A}$ (and gadget matrix with $\boldsymbol{W}_1, \dots, \boldsymbol{W}_\ell$)

$$\boldsymbol{A} \leftarrow \mathbb{Z}_q^{n \times m}, \quad \boldsymbol{A}_i := \boldsymbol{A}$$

$$\boldsymbol{W}_1, \dots, \boldsymbol{W}_\ell \leftarrow \mathbb{Z}_q^{n \times n}$$

$$\boldsymbol{B}_\ell = \left[\begin{array}{ccc|c} \boldsymbol{A} & & & \boldsymbol{W}_1 \\ & \ddots & & \vdots \\ & & \boldsymbol{A} & \boldsymbol{W}_\ell \end{array}\right]$$

To commit to an input $\boldsymbol{x} \in \{0,1\}^\ell$:

Use trapdoor for $\boldsymbol{B}_\ell$ to jointly sample $\boldsymbol{V}_1, \dots, \boldsymbol{V}_\ell, \widehat{\boldsymbol{C}}$ that satisfy

$$\left[\begin{array}{ccc|c} \boldsymbol{A} & & & \boldsymbol{W}_1 \\ & \ddots & & \vdots \\ & & \boldsymbol{A} & \boldsymbol{W}_\ell \end{array}\right] \cdot \begin{bmatrix} \boldsymbol{V}_1 \\ \vdots \\ \boldsymbol{V}_\ell \\ \boldsymbol{C} \end{bmatrix} = \begin{bmatrix} -x_1 \boldsymbol{G} \\ \vdots \\ -x_\ell \boldsymbol{G} \end{bmatrix}$$

Functional Commitments for Circuits

Commitment relation:

$$\begin{bmatrix} \boldsymbol{A} & & & \boldsymbol{W}_1 \\ & \ddots & & \vdots \\ & & \boldsymbol{A} & \boldsymbol{W}_\ell \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{V}_1 \\ \vdots \\ \boldsymbol{V}_\ell \\ \boldsymbol{C} \end{bmatrix} = \begin{bmatrix} -x_1 \boldsymbol{G} \\ \vdots \\ -x_\ell \boldsymbol{G} \end{bmatrix}$$

Homomorphic evaluation:

$$\boldsymbol{C}_1 = \boldsymbol{A}\boldsymbol{V}_1 + x_1 \boldsymbol{G}$$
$$\vdots$$
$$\boldsymbol{C}_\ell = \boldsymbol{A}\boldsymbol{V}_\ell + x_\ell \boldsymbol{G}$$

$\Rightarrow$

$$\boldsymbol{C}_f = \boldsymbol{A}\boldsymbol{V}_f + f(\boldsymbol{x}) \cdot \boldsymbol{G}$$

for all $i \in [\ell]$

$$\boldsymbol{A}\boldsymbol{V}_i + \boldsymbol{W}_i \boldsymbol{C} = -x_i \boldsymbol{G}$$

rearranging

$$-\boldsymbol{W}_i \boldsymbol{C} = \boldsymbol{A}\boldsymbol{V}_i + x_i \boldsymbol{G}$$

function of just the commitment $\boldsymbol{C}$

$$\widetilde{\boldsymbol{C}}_i = -\boldsymbol{W}_i \boldsymbol{C}$$

$$\widetilde{\boldsymbol{C}}_i = \boldsymbol{A}\boldsymbol{V}_i + x_i \boldsymbol{G}$$

Functional Commitments for Circuits

Commitment relation:

$$\begin{bmatrix} \mathbf{A} & & & \mathbf{W}_1 \\ & \ddots & & \vdots \\ & & \mathbf{A} & \mathbf{W}_\ell \end{bmatrix} \cdot \begin{bmatrix} \mathbf{V}_1 \\ \vdots \\ \mathbf{V}_\ell \\ \mathbf{C} \end{bmatrix} = \begin{bmatrix} -x_1 \mathbf{G} \\ \vdots \\ -x_\ell \mathbf{G} \end{bmatrix}$$

Homomorphic evaluation:

$$\mathbf{C}_1 = \mathbf{A}\mathbf{V}_1 + x_1 \mathbf{G}$$
$$\vdots$$
$$\mathbf{C}_\ell = \mathbf{A}\mathbf{V}_\ell + x_\ell \mathbf{G}$$

$$\Rightarrow \quad \mathbf{C}_f = \mathbf{A}\mathbf{V}_f + f(\mathbf{x}) \cdot \mathbf{G}$$

function of just the commitment $\mathbf{C}$

$$\widetilde{\mathbf{C}}_i = -\mathbf{W}_i \mathbf{C}$$

$$\widetilde{\mathbf{C}}_i = \mathbf{A}\mathbf{V}_i + x_i \mathbf{G}$$

$\widetilde{\mathbf{C}}_i$ is an encoding of x_i with randomness $\mathbf{V}_i$

compute on $\widetilde{\mathbf{C}}_1, \dots \widetilde{\mathbf{C}}_f$ ↓ ↓ compute on $\mathbf{V}_1, \dots, \mathbf{V}_\ell$

$$\widetilde{\mathbf{C}}_f = \mathbf{A}\mathbf{V}_{f,f(\mathbf{x})} + f(\mathbf{x})\mathbf{G}$$

$\widetilde{\mathbf{C}}_f$ is an encoding of $f(\mathbf{x})$ with randomness $\mathbf{V}_{f,f(\mathbf{x})}$

[GVW15]: independent $\mathbf{V}_i$ is sampled for each input bit, so commitments $\mathbf{C}_i$ are independent

- long commitment, security from SIS

[WW23a, WW23b]: publish a trapdoor that allows deriving $\mathbf{C}_i$ (and associated $\mathbf{V}_i$) from a **single** commitment $\widehat{\mathbf{C}}$

- short commitment, stronger assumption

Functional Commitments for Circuits

Commitment relation:

$$\left[\begin{array}{ccc|c} \boldsymbol{A} & & & \boldsymbol{W}_1 \\ & \ddots & & \vdots \\ & & \boldsymbol{A} & \boldsymbol{W}_\ell \end{array}\right] \cdot \begin{bmatrix} \boldsymbol{V}_1 \\ \vdots \\ \boldsymbol{V}_\ell \\ \boldsymbol{C} \end{bmatrix} = \begin{bmatrix} -x_1 \boldsymbol{G} \\ \vdots \\ -x_\ell \boldsymbol{G} \end{bmatrix}$$

Homomorphic evaluation:

$$\begin{matrix} \boldsymbol{C}_1 = \boldsymbol{A}\boldsymbol{V}_1 + x_1 \boldsymbol{G} \\ \vdots \\ \boldsymbol{C}_\ell = \boldsymbol{A}\boldsymbol{V}_\ell + x_\ell \boldsymbol{G} \end{matrix} \Rightarrow \boldsymbol{C}_f = \boldsymbol{A}\boldsymbol{V}_f + f(\boldsymbol{x}) \cdot \boldsymbol{G}$$

Opening is $\boldsymbol{V}_{f,f(x)}$ is
(short) linear function of $\boldsymbol{V}_1, \dots, \boldsymbol{V}_\ell$

Opening to function f proceeds exactly as in [GVW15]

To verify:

1. Expand commitment

$$\boldsymbol{C} \xrightarrow{\widetilde{\boldsymbol{C}}_i = -\boldsymbol{W}_i \boldsymbol{C}} \begin{matrix} \widetilde{\boldsymbol{C}}_1 = \boldsymbol{A}\boldsymbol{V}_1 + x_1 \boldsymbol{G} \\ \vdots \\ \widetilde{\boldsymbol{C}}_\ell = \boldsymbol{A}\boldsymbol{V}_\ell + x_\ell \boldsymbol{G} \end{matrix}$$

2. Homomorphically evaluate f

$$\widetilde{\boldsymbol{C}}_1, \dots \widetilde{\boldsymbol{C}}_\ell \Rightarrow \widetilde{\boldsymbol{C}}_f$$

3. Check verification relation

$$\boldsymbol{A}\boldsymbol{V}_{f,z} = \widetilde{\boldsymbol{C}}_f - z \cdot \boldsymbol{G}$$

Functional Commitments from Lattices

Security follows from ℓ-succinct SIS assumption [Wee23]:

SIS is hard with respect to $\boldsymbol{A}$ given a trapdoor (a basis) for the matrix

$$\boldsymbol{B}_\ell = \left[\begin{array}{ccc|c} \boldsymbol{A} & & & \boldsymbol{W}_1 \\ & \ddots & & \vdots \\ & & \boldsymbol{A} & \boldsymbol{W}_\ell \end{array}\right]$$

where $\boldsymbol{A} \leftarrow \mathbb{Z}_q^{n \times m}$ and $\boldsymbol{W}_i \leftarrow \mathbb{Z}_q^{n \times m}$

Falsifiable assumption but does not appear to reduce to standard SIS

$\ell = 1$ case does follow from plain SIS (and when $\boldsymbol{W}_i$ is <u>very</u> wide)

Open problem: Understanding security or attacks when $\ell > 1$

Functional Commitments from Lattices

Common reference string (for inputs of length ℓ):

matrices $\boldsymbol{A}_1, \boldsymbol{W}_1, \ldots, \boldsymbol{W}_\ell \in \mathbb{Z}_q^{n \times m}$

auxiliary data: trapdoor for $\boldsymbol{B}_\ell = \begin{bmatrix} \boldsymbol{A} & & & \boldsymbol{W}_1 \\ & \ddots & & \vdots \\ & & \boldsymbol{A} & \boldsymbol{W}_\ell \end{bmatrix}$

To commit to a vector $\boldsymbol{x} \in \{0,1\}^\ell$: sample $(\boldsymbol{V}_1, \ldots, \boldsymbol{V}_\ell, \boldsymbol{C})$

$$\begin{bmatrix} \boldsymbol{A} & & & \boldsymbol{W}_1 \\ & \ddots & & \vdots \\ & & \boldsymbol{A} & \boldsymbol{W}_\ell \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{V}_1 \\ \vdots \\ \boldsymbol{V}_\ell \\ \boldsymbol{C} \end{bmatrix} = \begin{bmatrix} -x_1 \boldsymbol{G} \\ \vdots \\ -x_\ell \boldsymbol{G} \end{bmatrix}$$

Commitment is $\boldsymbol{C}$

Openings for function f is $[\boldsymbol{V}_1 \mid \cdots \mid \boldsymbol{V}_\ell] \cdot \boldsymbol{H}_{\widetilde{\boldsymbol{C}}, f, \boldsymbol{x}}$

Scheme supports functions computable by Boolean circuits of (bounded) depth d

$$|\mathrm{crs}| = \ell^2 \cdot \mathrm{poly}(\lambda, d, \log \ell)$$

$$|\boldsymbol{C}| = \mathrm{poly}(\lambda, d, \log \ell)$$

$$\left|\boldsymbol{V}_{f, f(\boldsymbol{x})}\right| = \mathrm{poly}(\lambda, d, \log \ell)$$

Verification **time** scales with $|f|$ (i.e., size of circuit computing f)

Summary of Functional Commitments

New methodology for constructing lattice-based commitments:

1. Write down the main verification relation ($\boldsymbol{c} = \boldsymbol{A}_i \boldsymbol{v}_i + x_i \boldsymbol{t}_i$)
2. Publish a trapdoor for the linear system by the verification relation

Security analysis relies on new q-type variants of SIS:

*SIS with respect to $\boldsymbol{A}$ is hard given a trapdoor for a **related** matrix $\boldsymbol{B}$*

"Random" variant of the assumption implies vector commitments and reduces to SIS

"Structured" variant (ℓ-succinct SIS) implies functional commitments for circuits

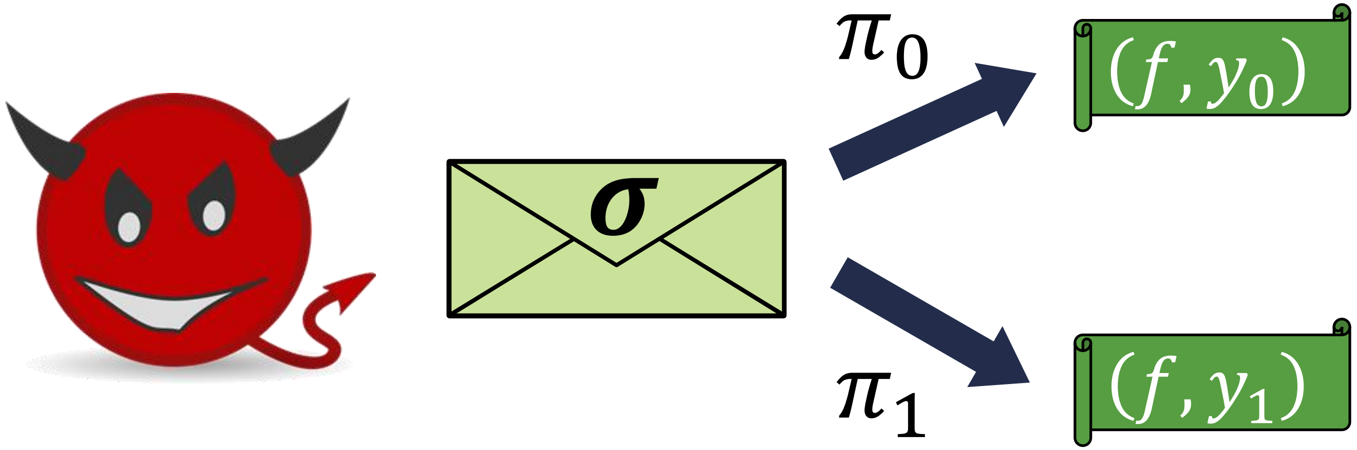
- Structure also enables **aggregating** openings

[see paper for details]

Cryptanalysis of Lattice-Based Knowledge Assumptions

Extractable Functional Commitments

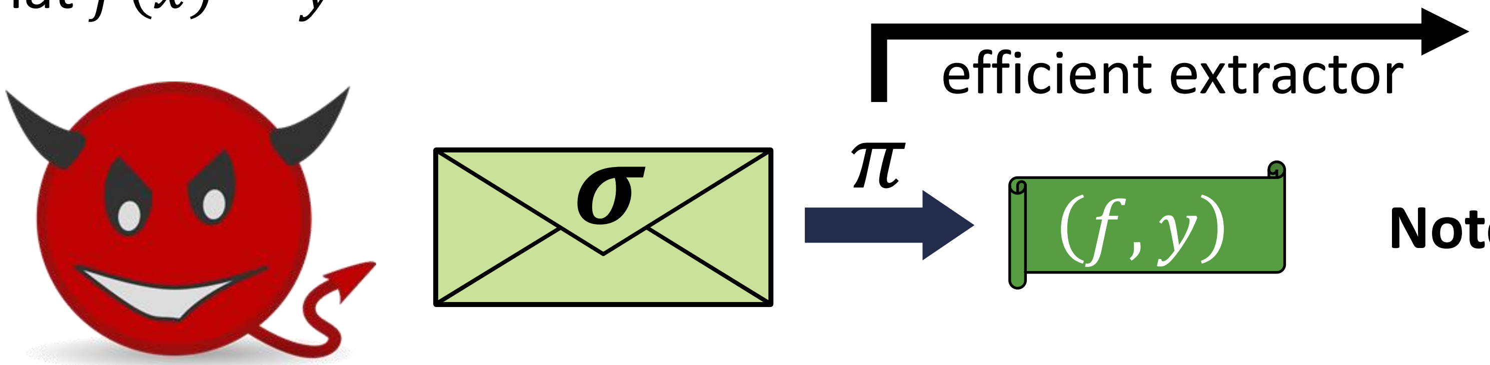
Binding: efficient adversary cannot open σ to two different values with respect to the **same** f



$\text{Verify}(\text{crs}, \sigma, (f, y_0), \pi_0) = 1$

$\text{Verify}(\text{crs}, \sigma, (f, y_1), \pi_1) = 1$

Extractability: efficient adversary that opens σ to y with respect to f must **know** an x such that $f(x) = y$



x such that $y = f(x)$

Note: f could have multiple outputs

Cryptanalysis of Lattice-Based Knowledge Assumptions

Typical lattice-based knowledge assumption (to get extractable commitments / SNARKs):

given (tall) matrices $\boldsymbol{A}, \boldsymbol{D}$ and short preimages $\boldsymbol{Z}$ of a random target $\boldsymbol{T}$

the only way an adversary can produce a short vector $\boldsymbol{v}$ such that $\boldsymbol{A}\boldsymbol{v}$ is in the image of $\boldsymbol{D}$ (i.e., $\boldsymbol{A}\boldsymbol{v} = \boldsymbol{D}\boldsymbol{c}$) is by setting $\boldsymbol{v} = \boldsymbol{Z}\boldsymbol{x}$

Observe: $\boldsymbol{A}\boldsymbol{v}$ for a random (short) $\boldsymbol{v}$ is outside the image of $\boldsymbol{D}$ (since $\boldsymbol{D}$ is tall)

Cryptanalysis of Lattice-Based Knowledge Assumptions

Typical lattice-based knowledge assumption (to get extractable commitments / SNARKs):

For extractable functional commitments:

- $\boldsymbol{Z}$ is in the CRS
- Commitment is $\boldsymbol{c} = \boldsymbol{D}\boldsymbol{T}\boldsymbol{x}$
- Opening is $\boldsymbol{v}$ where $\boldsymbol{A}\boldsymbol{v} = \boldsymbol{D}\boldsymbol{c}$

Extractable since valid opening can be associated with an honestly-generated commitment

given (tall) matrices $\boldsymbol{A}, \boldsymbol{D}$ and short preimages $\boldsymbol{Z}$ of a random target $\boldsymbol{T}$

the only way an adversary can produce a short vector $\boldsymbol{v}$ such that $\boldsymbol{A}\boldsymbol{v}$ is in the image of $\boldsymbol{D}$ (i.e., $\boldsymbol{A}\boldsymbol{v} = \boldsymbol{D}\boldsymbol{c}$) is by setting $\boldsymbol{v} = \boldsymbol{Z}\boldsymbol{x}$

Observe: $\boldsymbol{A}\boldsymbol{v}$ for a random (short) $\boldsymbol{v}$ is outside the image of $\boldsymbol{D}$ (since $\boldsymbol{D}$ is tall)

Obliviously Sampling a Solution

Typical lattice-based knowledge assumption (to get extractable commitments / SNARKs):

This work: algorithm to obliviously sample a solution $\boldsymbol{Av} = \boldsymbol{Dc}$ without knowledge of a linear combination $\boldsymbol{v} = \boldsymbol{Zx}$

Rewrite $\boldsymbol{AZ} = \boldsymbol{DT}$ as

$$[\boldsymbol{A} \mid \boldsymbol{DG}] \cdot \begin{bmatrix} \boldsymbol{Z} \\ -\boldsymbol{G}^{-1}(\boldsymbol{T}) \end{bmatrix} = \boldsymbol{0}$$

If $\boldsymbol{Z}$ and $\boldsymbol{T}$ are wide enough, we (heuristically) obtain a basis for $[\boldsymbol{A} \mid \boldsymbol{DG}]$

Obliviously Sampling a Solution

This work: algorithm to obliviously sample a solution $\boldsymbol{Av} = \boldsymbol{Dc}$ without knowledge of a linear combination $\boldsymbol{v} = \boldsymbol{Zx}$

Rewrite $\boldsymbol{AZ} = \boldsymbol{DT}$ as

$$[\boldsymbol{A} \mid \boldsymbol{DG}] \cdot \underbrace{\begin{bmatrix} \boldsymbol{Z} \\ -\boldsymbol{G}^{-1}(\boldsymbol{T}) \end{bmatrix}}_{\boldsymbol{B}^*} = \boldsymbol{0}$$

If $\boldsymbol{Z}$ and $\boldsymbol{T}$ are wide enough, we (heuristically) obtain a basis for $[\boldsymbol{A} \mid \boldsymbol{DG}]$

Oblivious sampler (Babai rounding):

1. Take any (non-zero) integer solution $\boldsymbol{y}$ where $[\boldsymbol{A} \mid \boldsymbol{DG}]\boldsymbol{y} = \boldsymbol{0} \bmod q$
2. Assuming $\boldsymbol{B}^*$ is full-rank over $\mathbb{Q}$, find $\boldsymbol{z}$ such that $\boldsymbol{B}^*\boldsymbol{z} = \boldsymbol{y}$ (over $\mathbb{Q}$)
3. Set $\boldsymbol{y}^* = \boldsymbol{y} - \boldsymbol{B}^*\lfloor \boldsymbol{z} \rceil = \boldsymbol{B}^*(\boldsymbol{z} - \lfloor \boldsymbol{z} \rceil)$ and parse into $\boldsymbol{v}, \boldsymbol{c}$

Correctness: $[\boldsymbol{A} \mid \boldsymbol{DG}] \cdot \boldsymbol{y}^* = [\boldsymbol{A} \mid \boldsymbol{DG}] \cdot \boldsymbol{B}^*(\boldsymbol{z} - \lfloor \boldsymbol{z} \rceil) = \boldsymbol{0} \bmod q$ and $\boldsymbol{y}^*$ is short

Obliviously Sampling a Solution

This work: algorithm to obliviously sample a solution $\boldsymbol{Av} = \boldsymbol{Dc}$ without knowledge of a linear combination $\boldsymbol{v} = \boldsymbol{Zx}$

Rewrite $\boldsymbol{AZ} = \boldsymbol{DT}$ as

$$[\boldsymbol{A} \mid \boldsymbol{DG}] \cdot \underbrace{\begin{bmatrix} \boldsymbol{Z} \\ -\boldsymbol{G}^{-1}(\boldsymbol{T}) \end{bmatrix}}_{\boldsymbol{B}^*} = \boldsymbol{0}$$

If $\boldsymbol{Z}$ and $\boldsymbol{T}$ are wide enough, we (heuristically) obtain a basis for $[\boldsymbol{A} \mid \boldsymbol{DG}]$

This solution is obtained by "rounding" off a long solution

Question: Can we explain such solutions as taking a short linear combination of $\boldsymbol{Z}$ (i.e., what the knowledge assumption asserts)

Oblivious sampler (Babai roun

1. Take any (non-zero) inte
2. Assuming $\boldsymbol{B}^*$ is full-rank
3. Set $\boldsymbol{y}^* = \boldsymbol{y} - \boldsymbol{B}^*\lfloor \boldsymbol{z} \rceil = \boldsymbol{B}$

Correctness: $[\boldsymbol{A} \mid \boldsymbol{DG}] \cdot \boldsymbol{y}^* = [\boldsymbol{A} \mid \boldsymbol{DG}] \cdot \boldsymbol{B}^*(\boldsymbol{z} - \lfloor \boldsymbol{z} \rceil) = \boldsymbol{0} \bmod q$ and $\boldsymbol{y}^*$ is short

Template for Analyzing Lattice-Based Knowledge Assumptions

1. Start with the key verification relation (i.e., knowledge of a short solution to a linear system)
2. Express verification relation as finding non-zero vector in the kernel of a lattice defined by the verification equation
3. Use components in the CRS to derive a basis for the related lattice

①

$$\boldsymbol{A}\boldsymbol{v} = \boldsymbol{D}\boldsymbol{c}$$

②

$$[\boldsymbol{A} \mid \boldsymbol{D}\boldsymbol{G}] \begin{bmatrix} \boldsymbol{v} \\ -\boldsymbol{G}^{-1}(\boldsymbol{c}) \end{bmatrix} = \boldsymbol{0}$$

③

$$[\boldsymbol{A} \mid \boldsymbol{D}\boldsymbol{G}] \cdot \begin{bmatrix} \boldsymbol{Z} \\ -\boldsymbol{G}^{-1}(\boldsymbol{T}) \end{bmatrix} = \boldsymbol{0}$$

Template for Analyzing Lattice-Based Knowledge Assumptions

1. Start with the key verification relation (i.e., knowledge of a short solution to a linear system)
2. Express verification relation as finding non-zero vector in the kernel of a lattice defined by the verification equation
3. Use components in the CRS to derive a basis for the related lattice

Implications:

- Oblivious sampler for integer variant of knowledge k-R-ISIS assumption from [ACLMT22]
  Implementation by Martin Albrecht: https://gist.github.com/malb/7c8b86520c675560be62eda98dab2a6f
- Breaks extractability of the (integer variant of the) linear functional commitment from [ACLMT22] assuming hardness of inhomogeneous SIS (i.e., existence of efficient extractor for oblivious sampler implies algorithm for inhomogeneous SIS)

Open question: Can we extend the attacks to break soundness of the SNARK?

Template for Analyzing Lattice-Based Knowledge Assumptions

1. Start with the key verification relation (i.e., knowledge of a short solution to a linear system)
2. Express verification relation as finding non-zero vector in the kernel of a lattice defined by the verification equation
3. Use components in the CRS to derive a basis for the related lattice

Implications:

- Oblivious sampler for intege
  Implementation by Martin Albrec
- Breaks extractability of the (
  [ACLMT22] assuming hardn
  for oblivious sampler implies algorithm for inhomogeneous SIS)

The SNARK considers extractable commitment for **quadratic** functions while our current oblivious sampler only works for **linear** functions in the case of [ACLMT22]

Open question: Can we extend the attacks to break soundness of the SNARK?

Open Questions

Understanding the hardness of ℓ-succinct SIS (hardness reductions or cryptanalysis)?

(Black-box) functional commitments with fast verification from standard SIS?

Cryptanalysis of lattice-based SNARKs based on knowledge k-R-ISIS [ACLMT22, CLM23, FLV23]

Our oblivious sampler (heuristically) falsifies the assumption, but does not break existing constructions

Formulation of new lattice-based knowledge assumptions that avoids our attacks

Thank you!