Lattice-Based Functional Commitments: Constructions and Cryptanalysis

David Wu
December 2023

based on joint work with Hoeteck Wee
Functional Commitments

\( x \) → Commit → "commitment" → Open + Verify → "opening"
Functional Commitments

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

Takes a common reference string and commits to an input \(x\)

Outputs commitment \(\sigma\) and commitment state \(st\)
Functional Commitments

\[
\sigma 
\]

\[
\text{Open + Verify} 
\]

\[
\pi
\]

\[
f(x)
\]

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

Open(st, \(f\)) \(\rightarrow \pi\)

Takes the commitment state \text{and a function} \(f\) and outputs an opening \(\pi\)

Verify(crs, \(\sigma, (f, y), \pi\)) \(\rightarrow 0/1\)

Checks whether \(\pi\) is valid opening of \(\sigma\) to value \(y\) with respect to \(f\)
**Functional Commitments**

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

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

- $\pi_1 \rightarrow (f, y_1)$
  - $\text{Verify}(\text{crs}, \sigma, (f, y_1), \pi_1) = 1$
**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:
\[ [x_1, x_2, \ldots, x_n] \]
commit to a vector, open at an index
\[ \text{ind}_i(x_1, \ldots, x_n) := x_i \]

Polynomial commitments:
\[ [\alpha_0, \alpha_1, \ldots, \alpha_d] \]
commit to a polynomial, open to the evaluation at \( x \)
\[ f_x(\alpha_0, \ldots, \alpha_d) := \alpha_0 + \alpha_1 x + \ldots + \alpha_d x^d \]
# Succinct Functional Commitments

*(not an exhaustive list!)*

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Function Class</th>
<th>Assumption</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Mer87]</td>
<td>vector commitment</td>
<td>collision-resistant hash functions</td>
</tr>
<tr>
<td>[LY10, CF13, LM19, GRWZ20]</td>
<td>vector commitment</td>
<td>$q$-type pairing assumptions</td>
</tr>
<tr>
<td>[CF13, LM19, BBF19]</td>
<td>vector commitment</td>
<td>groups of unknown order</td>
</tr>
<tr>
<td>[PPS21]</td>
<td>vector commitment</td>
<td>short integer solutions (SIS)</td>
</tr>
<tr>
<td>[KZG10, Lee20]</td>
<td>polynomial commitment</td>
<td>$q$-type pairing assumptions</td>
</tr>
<tr>
<td>[BFS19, BHRRS21, BF23]</td>
<td>polynomial commitment</td>
<td>groups of unknown order</td>
</tr>
<tr>
<td>[LRY16]</td>
<td>linear functions</td>
<td>$q$-type pairing assumptions</td>
</tr>
<tr>
<td>[ACLMT22]</td>
<td>constant-degree polynomials</td>
<td>$k$-R-ISIS assumption (falsifiable)</td>
</tr>
<tr>
<td>[LRY16]</td>
<td>Boolean circuits</td>
<td>collision-resistant hash functions + SNARKs</td>
</tr>
<tr>
<td>[dCP23]</td>
<td>Boolean circuits</td>
<td>SIS (non-succinct openings in general)</td>
</tr>
<tr>
<td>[KLVW23]</td>
<td>Boolean circuits</td>
<td>LWE (via batch arguments)</td>
</tr>
<tr>
<td>[BCFL23]</td>
<td>Boolean circuits</td>
<td>twin $k$-R-ISIS</td>
</tr>
<tr>
<td>[WW23a, WW23b]</td>
<td>Boolean circuits</td>
<td>$\ell$-succinct SIS</td>
</tr>
</tbody>
</table>
Framework for Lattice Commitments

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

Common reference string (for inputs of length $\ell$):

- matrices $A_1, ..., A_\ell \in \mathbb{Z}_q^{n \times m}$
- target vectors $t_1, ..., t_\ell \in \mathbb{Z}_q^n$

**auxiliary data:** cross-terms $u_{ij} \leftarrow A_i^{-1}(t_j) \in \mathbb{Z}_q^m$ where $i \neq j$

short (i.e., low-norm) vector satisfying $A_i u_{ij} = t_j$
Framework for Lattice Commitments

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

Common reference string (for inputs of length $\ell$):
- matrices $A_1, \ldots, A_\ell \in \mathbb{Z}_q^{n \times m}$
- target vectors $t_1, \ldots, t_\ell \in \mathbb{Z}_q^n$
- auxiliary data: cross-terms $u_{ij} \leftarrow A_i^{-1}(t_j) \in \mathbb{Z}_q^m$ where $i \neq j$

Commitment to $x \in \mathbb{Z}_q^\ell$:
\[
    c = \sum_{i \in [\ell]} x_i t_i
\]

Linear combination of target vectors

Opening to value $y$ at index $i$:
- short $v_i$ such that $c = A_i v_i + y \cdot t_i$

Honest opening:
\[
    v_i = \sum_{j \neq i} x_j u_{ij}
\]
\[
    A_i v_i + x_i t_i = \sum_{j \neq i} x_j A_i u_{ij} + x_i t_i = \sum_{j \in [\ell]} x_j t_j = c
\]

Correct as long as $x$ is short
Framework for Lattice Commitments

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

Common reference string (for inputs of length $\ell$):
- matrices $A_1, \ldots, A_\ell \in \mathbb{Z}_q^{n \times m}$
- target vectors $t_1, \ldots, t_\ell \in \mathbb{Z}_q^n$

auxiliary data: cross-terms $u_{ij} \leftarrow A_i^{-1}(t_j) \in \mathbb{Z}_q^m$ where $i \neq j$

[PPS21]: $A_i \leftarrow \mathbb{Z}_q^{n \times m}$ and $t_i \leftarrow \mathbb{Z}_q^n$ are independent and uniform

suffices for vector commitments (from SIS)

[ACLMT21]: $A_i = W_iA$ and $t_i = W_iu_i$ where $W_i \leftarrow \mathbb{Z}_q^{n \times n}, A \leftarrow \mathbb{Z}_q^{n \times m}, u_i \leftarrow \mathbb{Z}_q^n$

(one candidate adaptation to the integer case)

generalizes 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:** \( c = A_i v_i + x_i t_i \quad \forall i \in [\ell] \)

\( I_n \) denotes the identity matrix

\[
\begin{bmatrix}
A_1 \\
\vdots \\
A_\ell
\end{bmatrix}
\begin{bmatrix}
\vdots \\
-I_n \\
\vdots
\end{bmatrix}
\begin{bmatrix}
v_1 \\
\vdots \\
v_\ell
\end{bmatrix}
= 
\begin{bmatrix}
x_1 t_1 \\
\vdots \\
x_\ell t_\ell
\end{bmatrix}
\]

**Our approach:** rewrite \( \ell \) relations as a single linear system

\( I_n \) denotes the identity matrix
Our Approach

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

**Verification invariant:** $c = A_i v_i + x_i t_i \quad \forall i \in [\ell]

for a short $v_i$

**Our approach:** rewrite $\ell$ relations as a single linear system

$$
\begin{bmatrix}
A_1 & -G \\
\vdots & \vdots \\
A_\ell & -G
\end{bmatrix}
\begin{bmatrix}
v_1 \\
\vdots \\
v_\ell
\end{bmatrix}
= 
\begin{bmatrix}
-x_1 t_1 \\
\vdots \\
-x_\ell t_\ell
\end{bmatrix}
$$

"powers of two matrix"

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

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

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

**Verification invariant:** \( c = A_i v_i + x_i t_i \quad \forall i \in [\ell] \)

for a short \( v_i \)

Our approach: rewrite \( \ell \) relations as a single linear system

\[
\begin{bmatrix}
A_1 & \cdots & -G \\
\vdots & \ddots & \vdots \\
A_\ell & \cdots & -G \\
\end{bmatrix}
\begin{bmatrix}
v_1 \\
\vdots \\
v_\ell \\
\end{bmatrix}
= 
\begin{bmatrix}
-x_1 t_1 \\
\vdots \\
-x_\ell t_\ell \\
\end{bmatrix}
\]

Common reference string:
- matrices \( A_1, \ldots, A_\ell \in \mathbb{Z}_q^{n \times m} \)
- target vectors \( t_1, \ldots, t_\ell \in \mathbb{Z}_q^m \)

auxiliary data: cross-terms \( u_{ij} \leftarrow A_i^{-1}(t_j) \)
Our Approach

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

**Verification invariant:** $c = A_i v_i + x_i t_i \quad \forall i \in [\ell]

for a short $v_i$

**Our approach:** rewrite $\ell$ relations as a single linear system

$$\begin{bmatrix}
A_1 & \cdots & -G \\
\vdots & \ddots & \vdots \\
A_\ell & \cdots & -G
\end{bmatrix}
\begin{bmatrix}
v_1 \\
\vdots \\
v_\ell
\end{bmatrix}
= 
\begin{bmatrix}
-x_1 t_1 \\
\vdots \\
-x_\ell t_\ell
\end{bmatrix}
\hat{c}
$$

**Common reference string:**
- matrices $A_1, ..., A_\ell \in \mathbb{Z}_q^{n \times m}$
- target vectors $t_1, ..., t_\ell \in \mathbb{Z}_q^m$
- auxiliary data: cross-terms $u_{ij} = A_i^{-1}(t_j)$

Trapdoor for $B_\ell$ can be used to sample short solutions $x$ to the linear system $B_\ell x = y$ (for arbitrary $y$)
Our Approach

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

**Verification invariant:** \( c = A_i v_i + x_i t_i \quad \forall i \in [\ell] \)

for a short \( v_i \)

Our approach: rewrite \( \ell \) relations as a single linear system

\[
\begin{bmatrix}
A_1 & -G \\
\vdots & \vdots \\
A_\ell & -G
\end{bmatrix}
\begin{bmatrix}
v_1 \\
\vdots \\
v_\ell
\end{bmatrix}
= \begin{bmatrix}
-x_1 t_1 \\
\vdots \\
-x_\ell t_\ell
\end{bmatrix}
\]

\( B_\ell \)

Committing to an input \( x \): Use trapdoor for \( B_\ell \) to jointly sample a solution \( v_1, ..., v_\ell, \hat{c} \)

\( c = G\hat{c} \) is the commitment and \( v_1, ..., v_\ell \) 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: \( c = A_i v_i + x_i t_i \quad \forall i \in [\ell] \)
for a short \( v_i \)

Our approach: rewrite \( \ell \) relations as a single linear system

\[
\begin{bmatrix}
A_1 & \cdots & -G \\
\vdots & \ddots & \vdots \\
A_\ell & \cdots & -G
\end{bmatrix}
\begin{bmatrix}
v_1 \\
\vdots \\
v_\ell
\end{bmatrix}
\begin{bmatrix}
\hat{c} \\
\vdots \\
\hat{c}
\end{bmatrix}
= 
\begin{bmatrix}
-x_1 t_1 \\
\vdots \\
-x_\ell t_\ell
\end{bmatrix}
\]

Committing to an input \( x \):

Use trapdoor for \( B_\ell \) to jointly sample a solution \( v_1, ..., v_\ell, \hat{c} \)

\( c = G\hat{c} \) is the commitment and \( v_1, ..., v_\ell \) 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:** \( c = A_i v_i + x_i t_i \quad \forall i \in [\ell] \)
for a short \( v_i \)

Suppose adversary can break binding

outputs \( c, (v_i, x_i), (v'_i, x'_i) \) such that

\[
\begin{align*}
c & = A_i v_i + x_i t_i \\
\end{align*}
\]

\[
\begin{align*}
= A_i v'_i + x'_i t_i \\
\end{align*}
\]

given matrices \( A_1, \ldots, A_\ell \)
target vectors \( t_1, \ldots, t_\ell \)
trapdoor for \( B_\ell \)

Short integer solutions (SIS)
given \( A \leftarrow \mathbb{Z}_q^{n \times m} \), hard to find short \( x \neq 0 \) such that \( Ax = 0 \)

\[
A_i (v_i - v'_i) = (x_i - x'_i)e_1
\]

\( v_i - v'_i \) is a SIS solution for \( A_i \) without the first row

set \( A_i \leftarrow \mathbb{Z}_q^{n \times m} \)
set \( t_i = e_1 = [1,0,\ldots,0]^T \)
Proving Security

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

**Verification invariant:** \( c = A_i v_i + x_i t_i \quad \forall i \in [\ell] \)

for a short \( v_i \)

Adversary that breaks binding can solve SIS with respect to \( A_i \)

(*technically \( A_i \) without the first row – which is equivalent to SIS with dimension \( n - 1 \))

but... adversary also gets additional information beyond \( A_i \)

\[
B_\ell = \begin{bmatrix}
A_1 \\
\vdots \\
A_\ell
\end{bmatrix}
\begin{bmatrix}
-G \\
-\cdots\\
-\cdots
\end{bmatrix}
\]

Adversary sees **trapdoor** for \( B_\ell \)
Basis-Augmented SIS (BASIS) Assumption

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

**Verification invariant:** \( c = A_i v_i + x_i t_i \quad \forall i \in [\ell] \)

for a short \( v_i \)

Adversary that breaks binding can solve SIS with respect to \( A_i \)

Basis-augmented SIS (BASIS) assumption:

*SIS is hard with respect to \( A_i \)

given a trapdoor (a basis) for the matrix

\[
B_\ell = \begin{bmatrix}
A_1 & -G \\
\vdots & \vdots \\
A_\ell & -G
\end{bmatrix}
\]

Can simulate CRS from BASIS challenge:

matrices \( A_1, \ldots, A_\ell \leftarrow \mathbb{Z}_{q}^{n \times m} \)

 trapdoor for \( B_\ell \)
Basis-Augmented SIS (BASIS) Assumption

SIS is hard with respect to $A_i$ given a trapdoor (a basis) for the matrix

$$B_\ell = \begin{bmatrix} A_1 & \vdots & \vdots \\ \vdots & \ddots & \vdots \\ A_\ell & \vdots & -G \end{bmatrix}$$

When $A_1, \ldots, A_\ell \leftarrow \mathbb{Z}_q^{n \times m}$ 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 \( \ell \)):

matrices \( A_1, \ldots, A_\ell \in \mathbb{Z}_q^{n \times m} \)

auxiliary data: trapdoor for \( B_\ell = \begin{bmatrix} A_1 & -G \\ \vdots & \vdots \\ A_\ell & -G \end{bmatrix} \)

To commit to a vector \( x \in \mathbb{Z}_q^\ell \): sample solution \((v_1, \ldots, v_\ell, \hat{c})\)

\[
\begin{bmatrix} A_1 \\ \vdots \\ A_\ell \end{bmatrix} \cdot \begin{bmatrix} -G \\ \vdots \\ -G \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ \vdots \\ v_\ell \end{bmatrix} = \begin{bmatrix} -x_1 e_1 \\ \vdots \\ -x_\ell e_\ell \end{bmatrix}
\]

Commitment is \( c = G\hat{c} \)  
Openings are \( v_1, \ldots, v_\ell \)

Can commit and open to arbitrary \( \mathbb{Z}_q \) vectors

Commitments and openings statistically hide unopened components

Linearly homomorphic: \( c + c' \) is a commitment to \( x + x' \) with openings \( v_i + v'_i \)
Functional Commitments for Circuits

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

(f can be an arbitrary Boolean circuit)

Will need some basic lattice machinery for homomorphic computation

[GSW13, BGGHNSVV14, GVW15]

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

\[
C_1 = AV_1 + x_1 G \\
\vdots \\
C_\ell = AV_\ell + x_\ell G
\]

$C_i$ is an encoding of $x_i$ with (short) randomness $V_i$

$C_f$ is an encoding of $f(x)$ with (short) randomness $V_f$

$C_f = AV_f + f(x) \cdot G$
Replace random $A_i$ with a single $A$ (and gadget matrix with $W_1, \ldots, W_\ell$)

$A \leftarrow \mathbb{Z}_{q}^{n \times m}$, $A_i := A$

$W_1, \ldots, W_\ell \leftarrow \mathbb{Z}_{q}^{n \times n}$

Common reference string contains trapdoor for matrix $B_\ell$:

$$B_\ell = \begin{bmatrix} A & W_1 \\ \vdots & \vdots \\ A & W_\ell \end{bmatrix}$$
To commit to an input $x \in \{0,1\}^\ell$:

Use trapdoor for $B_\ell$ to **jointly** sample $V_1, ..., V_\ell, \widehat{C}$ that satisfy

\[
\begin{bmatrix}
A & W_1 \\
\vdots & \vdots \\
A & W_\ell
\end{bmatrix}
\begin{bmatrix}
V_1 \\
\vdots \\
V_\ell
\end{bmatrix}
= 
\begin{bmatrix}
-x_1 G \\
\vdots \\
-x_\ell G
\end{bmatrix}
\]
**Functional Commitments for Circuits**

Commitment relation:

\[
\begin{bmatrix}
A & W_1 \\
& \ddots & \ddots \\
& & A & W_\ell
\end{bmatrix}
\begin{bmatrix}
V_1 \\
\vdots \\
V_\ell
\end{bmatrix}
= \begin{bmatrix}
-x_1 G \\
\vdots \\
-x_\ell G
\end{bmatrix}
\]

Homomorphic evaluation:

- \( C_1 = AV_1 + x_1 G \)
- \( \vdots \)
- \( C_\ell = AV_\ell + x_\ell G \)
- \( C_f = AV_f + f(x) \cdot G \)

for all \( i \in [\ell] \)

\[
AV_i + W_i C = -x_i G
\]

rearranging

\[
-W_i C = AV_i + x_i G
\]
Functional Commitments for Circuits

Commitment relation:

\[
\begin{bmatrix}
A & W_1 \\
\vdots & \vdots \\
A & W_\ell
\end{bmatrix}
\begin{bmatrix}
V_1 \\
V_\ell \\
C
\end{bmatrix}
= \begin{bmatrix}
-x_1 G \\
\vdots \\
-x_\ell G
\end{bmatrix}
\]

Homomorphic evaluation:

\[
C_1 = AV_1 + x_1 G \\
\vdots \\
C_\ell = AV_\ell + x_\ell G
\]

\[
\bar{C}_f = AV_{f,f(x)} + f(x)G
\]

\[
\bar{C}_i = AV_i + x_i G
\]

\[
\bar{C}_i = -W_i C
\]

\[
\bar{C}_f = A \bar{V}_{f,f(x)} + f(x)G
\]

function of just the commitment \(C\)

compute on \(\bar{c}_1, \ldots, \bar{c}_f\)
compute on \(V_1, \ldots, V_\ell\)

\[\text{[GVW15]: independent } V_i \text{ is sampled for each input bit, so commitments } C_i \text{ are independent}
- \text{ long commitment, security from SIS}
\]

\[\text{[WW23a, WW23b]: publish a trapdoor that allows deriving } C_i \text{ (and associated } V_i) \text{ from a single commitment } \bar{C}]
- \text{ short commitment, stronger assumption}\]
Functional Commitments for Circuits

**Commitment relation:**

\[
\begin{bmatrix}
A & W_1 \\
\vdots & \vdots \\
A & W_\ell
\end{bmatrix}
\begin{bmatrix}
V_1 \\
\vdots \\
V_\ell \\
C
\end{bmatrix} =
\begin{bmatrix}
-x_1 G \\
\vdots \\
-x_\ell G
\end{bmatrix}
\]

**Homomorphic evaluation:**

\[
C_1 = AV_1 + x_1 G \\
\vdots \\
C_\ell = AV_\ell + x_\ell G
\]

Opening is \( V_{f,f}(x) \) is
(short) linear function of \( V_1, \ldots, V_\ell \)

Opening to function \( f \) proceeds exactly as in [GVW15]

**To verify:**

1. Expand commitment
   \[
   \tilde{C}_i = -w_i c
   \]
   \[
   \tilde{C}_1 = AV_1 + x_1 G \\
   \vdots \\
   \tilde{C}_\ell = AV_\ell + x_\ell G
   \]

2. Homomorphically evaluate \( f \)
   \[
   \tilde{C}_1, \ldots, \tilde{C}_\ell 
   \]
   \[
   \tilde{C}_f
   \]

3. Check verification relation
   \[
   AV_{f,z} = \tilde{C}_f - z \cdot G
   \]
Function of Commitments from Lattices

Security follows from $\ell$-succinct SIS assumption [Wee23]:

*SIS is hard with respect to $A$ given a trapdoor (a basis) for the matrix*

$$B_\ell = \begin{bmatrix} A & W_1 \\ \vdots & \vdots \\ A & W_\ell \end{bmatrix}$$

where $A \leftarrow \mathbb{Z}_{q}^{n \times m}$ and $W_i \leftarrow \mathbb{Z}_{q}^{n \times m}$

Falsifiable assumption but does not appear to reduce to standard SIS

$\ell = 1$ case does follow from plain SIS (and when $W_i$ is very wide)

**Open problem:** Understanding security or attacks when $\ell > 1$
Common reference string (for inputs of length $\ell$):

matrices $A_1, W_1, \ldots, W_\ell \in \mathbb{Z}_q^{n \times m}$

**auxiliary data:** trapdoor for $B_\ell = \begin{bmatrix} A & W_1 \\ \vdots & \vdots \\ A & W_\ell \end{bmatrix}$

To commit to a vector $x \in \{0,1\}^\ell$: sample $(V_1, \ldots, V_\ell, C)$

$$\begin{bmatrix} A \\ \vdots \\ A \end{bmatrix} \begin{bmatrix} W_1 \\ \vdots \\ W_\ell \end{bmatrix} \cdot \begin{bmatrix} V_1 \\ \vdots \\ V_\ell \\ C \end{bmatrix} = \begin{bmatrix} -x_1 G \\ \vdots \\ -x_\ell G \end{bmatrix}$$

Commitment is $C$

Openings for function $f$ is $[V_1 | \cdots | V_\ell] \cdot H_{\tilde{c},f,x}$

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

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

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

$|V_{f,f}(x)| = \text{poly}(\lambda, d, \log \ell)$

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 \( c = A_i v_i + x_i t_i \)
2. Publish a trapdoor for the linear system by the verification relation

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

\[
\text{SIS with respect to } A \text{ is hard given a trapdoor for a related matrix } B
\]

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

“Structured” variant (\( \ell \)-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 $\sigma$ to two different values with respect to the same $f$

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

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

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

$\sigma$  $\pi$  $(f, y)$   efficient extractor  $x$ 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 $A$, $D$ and short preimages $Z$ of a random target $T$

the only way an adversary can produce a short vector $v$ such that $Av$ is in the image of $D$ (i.e., $Av = Dc$) is by setting $v = Zx$

Observe: $Av$ for a random (short) $v$ is outside the image of $D$ (since $D$ is tall)
Typical lattice-based knowledge assumption (to get extractable commitments / SNARKs):

Given (tall) matrices $A$, $D$ and short preimages $Z$ of a random target $T$

The only way an adversary can produce a short vector $v$ such that $Av$ is in the image of $D$ (i.e., $Av = Dc$) is by setting $v = Zx$

Observe: $Av$ for a random (short) $v$ is outside the image of $D$ (since $D$ is tall)
Obliviously Sampling a Solution

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

\[ AX = DT \]

This work: algorithm to obliviously sample a solution \( Av = Dc \) without knowledge of a linear combination \( v = Zx \)

Rewrite \( AZ = DT \) as

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

If \( Z \) and \( T \) are wide enough, we (heuristically) obtain a basis for \([A \mid DG]\)
This work: algorithm to obliviously sample a solution $Av = Dc$ without knowledge of a linear combination $v = Zx$

Rewrite $AZ = DT$ as

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

If $Z$ and $T$ are wide enough, we (heuristically) obtain a basis for $[A | DG]$

Oblivious sampler (Babai rounding):

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

Correctness: $[A | DG] \cdot y^* = [A | DG] \cdot B^*(z - [z]) = 0 \mod q$ and $y^*$ is short
Obliviously Sampling a Solution

This work: algorithm to obliviously sample a solution \( A\mathbf{v} = D\mathbf{c} \) without knowledge of a linear combination \( \mathbf{v} = Z\mathbf{x} \)

Rewrite \( AZ = DT \) as

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

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

This solution is obtained by “rounding” off a long solution

Oblivious sampler (Babai rounding):

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

Correctness: \( [A \mid DG] \cdot \mathbf{y}^* = [A \mid DG] \cdot B^*(\mathbf{z} - \lfloor \mathbf{z} \rfloor) = 0 \mod q \) and \( \mathbf{y}^* \) is short

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

\[ Av = Dc \]  \quad \Rightarrow \quad \begin{bmatrix} A & DG \end{bmatrix} \begin{bmatrix} v \\ -G^{-1}(c) \end{bmatrix} = 0 \]  \quad \Rightarrow \quad \begin{bmatrix} A & DG \end{bmatrix} \cdot \begin{bmatrix} Z \\ -G^{-1}(T) \end{bmatrix} = 0 \]
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](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 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?
Open Questions

Understanding the hardness of \(\ell\)-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!