Zero-Knowledge Arguments for Matrix-Vector Relations and Lattice-Based Group Encryption

Benoît Libert[1] San Ling[2] Fabrice Mouhartem[1]
Khoa Nguyen[2] Huaxiong Wang[2]

[1]École Normale Supérieure de Lyon (France)

[2]Nanyang Technological University (Singapore)

ASIACRYPT 2016, Hanoi, Dec 5th 2016

Outline

1. Introduction
   - Group Encryption
   - Towards Realizing Lattice-Based Group Encryption
2. Our Results and Techniques
   - Proving “Quadratic Relations” in Zero-Knowledge

Group Signature and Group Encryption

- Group signature [CvH - EC'91]: Group member can anonymously sign messages on behalf of the whole group.
  $\Rightarrow$ Hiding the source of the messages within registered signers.

Group Signature and Group Encryption

- Group signature [CvH - EC'91]: Group member can anonymously sign messages on behalf of the whole group.
  $\Rightarrow$ Hiding the source of the messages within registered signers.

- Group encryption [KTY - AC'07]: the encryption analogue of group signature. Sender can encrypt messages to an anonymous group member.
  $\Rightarrow$ Hiding the destination of the messages within registered receivers.

Group Signature and Group Encryption

- Group signature [CvH - EC'91]: Group member can anonymously sign messages on behalf of the whole group.
  $\Rightarrow$ Hiding the source of the messages within registered signers.

- Group encryption [KTY - AC'07]: the encryption analogue of group signature. Sender can encrypt messages to an anonymous group member.
  $\Rightarrow$ Hiding the destination of the messages within registered receivers.

- Group members are kept accountable for their actions: an opening authority can un-anonymize the signatures/ciphertexts - should the needs arise.

Group Encryption [KTY - AC'07]

GE allows encrypting while proving that:

1. The ciphertext is well-formed and intended for some registered group member who will be able to decrypt;
2. The opening authority will be able identify the receiver if necessary;
3. The plaintext satisfies certain properties.

Group Encryption [KTY - AC'07]

GE allows encrypting while proving that:

1. The ciphertext is well-formed and intended for some registered group member who will be able to decrypt;
2. The opening authority will be able identify the receiver if necessary;
3. The plaintext satisfies certain properties.

Possible applications of GE:

- Firewall filtering
- Anonymous trusted third parties
- Cloud storage services
- Hierarchical group signatures [TW - ICALP'05].

Previous Works on Group Encryption

- [KTY - AC'07] introduced GE, and provided:
  - Modular design based on digital signatures, anonymous CCA-secure public-key encryption, interactive zero-knowledge proofs;
  - Concrete instantiation based on number-theoretic assumptions.

Previous Works on Group Encryption

- [KTY - AC'07] introduced GE, and provided:
  - Modular design based on digital signatures, anonymous CCA-secure public-key encryption, interactive zero-knowledge proofs;
  - Concrete instantiation based on number-theoretic assumptions.
- [CLY - AC'09]: non-interactive GE in the standard model under pairing-related assumptions.

Previous Works on Group Encryption

- [KTY - AC'07] introduced GE, and provided:
  - Modular design based on digital signatures, anonymous CCA-secure public-key encryption, interactive zero-knowledge proofs;
  - Concrete instantiation based on number-theoretic assumptions.
- [CLY - AC'09]: non-interactive GE in the standard model under pairing-related assumptions.
- [El Aimani,Joye - ACNS'13] suggested various improvements.

Previous Works on Group Encryption

- [KTY - AC'07] introduced GE, and provided:
  - Modular design based on digital signatures, anonymous CCA-secure public-key encryption, interactive zero-knowledge proofs;
  - Concrete instantiation based on number-theoretic assumptions.
- [CLY - AC'09]: non-interactive GE in the standard model under pairing-related assumptions.
- [El Aimani,Joye - ACNS'13] suggested various improvements.
- [LYJP - PKC'14]: refined traceability mechanism.

Previous Works on Group Encryption

- [KTY - AC'07] introduced GE, and provided:
  - Modular design based on digital signatures, anonymous CCA-secure public-key encryption, interactive zero-knowledge proofs;
  - Concrete instantiation based on number-theoretic assumptions.
- [CLY - AC'09]: non-interactive GE in the standard model under pairing-related assumptions.
- [El Aimani,Joye - ACNS'13] suggested various improvements.
- [LYJP - PKC'14]: refined traceability mechanism.

✗ All existing realizations of GE rely on number-theoretic assumptions.

? Construction from other assumptions, e.g., lattice-based?

In the World of Lattice-Based Crypto...

Many lattice-based group signatures published in the last 6 years.

- First constructions: [GKV - AC'10], [CNR - SCN'12] - linear-size signatures, static groups.
- Logarithmic-size signatures: [LLLS - AC'13].
- Improvements: [NZZ - PKC'15], [LNW - PKC'15], [LLNW - EC'16].
- With additional features: [LLNW - PKC'14], [LNW - ACNS'16].
- Dynamic groups: [LLMNW - AC'16].

In the World of Lattice-Based Crypto...

Many lattice-based group signatures published in the last 6 years.

- First constructions: [GKV - AC'10], [CNR - SCN'12] - linear-size signatures, static groups.
- Logarithmic-size signatures: [LLLS - AC'13].
- Improvements: [NZZ - PKC'15], [LNW - PKC'15], [LLNW - EC'16].
- With additional features: [LLNW - PKC'14], [LNW - ACNS'16].
- Dynamic groups: [LLMNW - AC'16].

But no lattice-based GE so far! Note that both GS and GE rely on

- Ordinary signatures;
- Public-key encryption;
- **Supporting zero-knowledge proofs**.

Where is the main technical difficulty?

Existing ZK Protocols in Lattice-Based Crypto

Two main classes:

1. Schnorr-like [Schnorr - Crypto'89] approach.
   - Introduced by Lyubashevsky [Lyu - PKC'08, EC'12]: *rejection sampling*.
2. Stern-like [Stern - Crypto'93, IEEE IT'96] approach.
   - First considered in the lattice setting by [KTX - AC'08].
   - Empowered by [LNSW - PKC'13]: *decomposition* and *extension*.

Existing ZK Protocols in Lattice-Based Crypto

Two main classes:

1. Schnorr-like [Schnorr - Crypto'89] approach.
   - Introduced by Lyubashevsky [Lyu - PKC'08, EC'12]: *rejection sampling*.
2. Stern-like [Stern - Crypto'93, IEEE IT'96] approach.
   - First considered in the lattice setting by [KTX - AC'08].
   - Empowered by [LNSW - PKC'13]: *decomposition* and *extension*.

These techniques deal with **linear relations**, i.e., equations containing terms:

$$\text{(public matrix)}\cdot\text{(secret vector)},$$

where the secret vector may satisfy some constraints (e.g., smallness).

- The (I)SIS relation [Ajtai - STOC'96, GPV - STOC'08]:
$$\mathbf{A}\cdot\mathbf{x} = \mathbf{u} \bmod q, \text{ for public } (\mathbf{A},\mathbf{u}).$$
- The LWE relation [Regev - STOC'05]:
$$\mathbf{A}\cdot\mathbf{s}+\mathbf{e} = \mathbf{b} \bmod q, \text{ for public } (\mathbf{A},\mathbf{b}).$$

The Case of Lattice-Based Group Signatures

A modular design for GS [BMW-EC'03]: **sign-then-encrypt-then-prove**

- Each user has a signature σ on his identity *id*, issued by the group manager (GM).
- In the process of generating GS, the user encrypts *id* to $\mathbf{c}$ - using the public key of the opening authority (OA), then proves in ZK that:

1. He has a secret valid pair (id, σ), w.r.t. pk_{GM}.
2. $\mathbf{c}$ is a well-formed ciphertext of *id*, w.r.t. pk_{OA}.

The Case of Lattice-Based Group Signatures

A modular design for GS [BMW-EC'03]: **sign-then-encrypt-then-prove**

- Each user has a signature σ on his identity *id*, issued by the group manager (GM).
- In the process of generating GS, the user encrypts *id* to $\mathbf{c}$ - using the public key of the opening authority (OA), then proves in ZK that:

1. He has a secret valid pair (id, σ), w.r.t. pk_{GM}.
2. $\mathbf{c}$ is a well-formed ciphertext of *id*, w.r.t. pk_{OA}.

✓ Known techniques allow to realize the core ZK components required by group signatures, for SIS-based signatures and LWE-based encryption.

Towards Realizing Lattice-Based Group Encryption

A modular design:

- Each member has a key pair (sk, pk) for an anonymous encryption scheme.
- Manager signs member's public key pk, and publishes (pk, σ).

Towards Realizing Lattice-Based Group Encryption

A modular design:

- Each member has a key pair (sk, pk) for an anonymous encryption scheme.
- Manager signs member's public key pk, and publishes (pk, σ).
- Sender uses pk to encrypt a message μ satisfying relation R, obtains $\mathbf{c}$.
- Sender also encrypts pk under the pk_{OA}, obtains $\mathbf{c}_{\mathrm{OA}}$.

Towards Realizing Lattice-Based Group Encryption

A modular design:

- Each member has a key pair (sk, pk) for an anonymous encryption scheme.
- Manager signs member's public key pk, and publishes (pk, σ).
- Sender uses pk to encrypt a message μ satisfying relation R, obtains $\mathbf{c}$.
- Sender also encrypts pk under the pk_{OA}, obtains $\mathbf{c}_{\mathrm{OA}}$.
- Prove that:

1. **$\mathbf{c}$ is a correct encryption of some message μ, w.r.t a hidden pk**;
2. Sender knows a valid signature σ on pk, w.r.t. pk_{GM}; $\mathbf{c}_{\mathrm{OA}}$ is a correct encryption of pk, w.r.t. pk_{OA}; The message μ satisfies relation R.

Towards Realizing Lattice-Based Group Encryption

A modular design:

- Each member has a key pair (sk, pk) for an anonymous encryption scheme.
- Manager signs member's public key pk, and publishes (pk, σ).
- Sender uses pk to encrypt a message μ satisfying relation R, obtains $\mathbf{c}$.
- Sender also encrypts pk under the pk_{OA}, obtains $\mathbf{c}_{\mathrm{OA}}$.
- Prove that:

1. **$\mathbf{c}$ is a correct encryption of some message μ, w.r.t a hidden pk**;
2. Sender knows a valid signature σ on pk, w.r.t. pk_{GM}; $\mathbf{c}_{\mathrm{OA}}$ is a correct encryption of pk, w.r.t. pk_{OA}; The message μ satisfies relation R.

Main Difficulty

We would have to handle an LWE relation with **hidden-but-certified** matrix:

$$\mathbf{X} \cdot \mathbf{s} + \mathbf{e} = \mathbf{b} \bmod q.$$

We call this **"quadratic relation"**: Main obstacle; new ideas are required.

Outline

Our Results

We introduce:

1. Zero-knowledge arguments for "quadratic relations", e.g.,

$$\mathbf{b} = \mathbf{X} \cdot \mathbf{s} + \mathbf{e} \bmod q,$$

where $\mathbf{X} \in \mathbb{Z}_q^{m \times n}$, $\mathbf{s} \in \mathbb{Z}_q^n$ may satisfy additional relations.

   - Approach: Developing Stern-like protocols, i.e., "linear $\rightarrow$ quadratic".
   - New techniques: May be of independent interest.

Our Results

We introduce:

1. Zero-knowledge arguments for "quadratic relations", e.g.,

$$\mathbf{b} = \mathbf{X} \cdot \mathbf{s} + \mathbf{e} \bmod q,$$

   where $\mathbf{X} \in \mathbb{Z}_q^{m \times n}$, $\mathbf{s} \in \mathbb{Z}_q^n$ may satisfy additional relations.

   - Approach: Developing Stern-like protocols, i.e., "linear $\rightarrow$ quadratic".
   - New techniques: May be of independent interest.

2. The first lattice-based group encryption scheme.

   - Under the LWE and SIS assumptions, the scheme is proven secure in the [KTY - AC'07] model.

Stern's Ideas

[Stern - '93,'96]: A zero-knowledge protocol for the syndrome decoding problem.

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{u} \bmod 2,$$

for public $(\mathbf{A}, \mathbf{u})$ and secret binary vector $\mathbf{x}$ having fixed Hamming weight w.

Stern's Ideas

[Stern - '93,'96]: A zero-knowledge protocol for the syndrome decoding problem.

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{u} \bmod 2,$$

for public $(\mathbf{A}, \mathbf{u})$ and secret binary vector $\mathbf{x}$ having fixed Hamming weight w.

Stern's Ideas

1. **Permuting:** Proving the witness constraint using random permutation.
   - Send the verifier $\pi(\mathbf{x})$.
   - $\mathbf{x}$ has constraint "binary vector with weight w" iff $\pi(\mathbf{x})$ does.

   The randomness of π protects the actual value of $\mathbf{x}$.

Stern's Ideas

[Stern - '93,'96]: A zero-knowledge protocol for the syndrome decoding problem.

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{u} \bmod 2,$$

for public $(\mathbf{A}, \mathbf{u})$ and secret binary vector $\mathbf{x}$ having fixed Hamming weight w.

Stern's Ideas

1. **Permuting:** Proving the witness constraint using random permutation.
   - Send the verifier $\pi(\mathbf{x})$.
   - $\mathbf{x}$ has constraint "binary vector with weight w" iff $\pi(\mathbf{x})$ does.

   The randomness of π protects the actual value of $\mathbf{x}$.
2. **Masking:** Proving the linear equation using a random masking $\mathbf{r}$.
   - Send the verifier $\mathbf{y} = \mathbf{x} + \mathbf{r}$, and show that: $\mathbf{A} \cdot \mathbf{y} = \mathbf{u} + \mathbf{A} \cdot \mathbf{r}$.

Stern's Ideas

[Stern - '93,'96]: A zero-knowledge protocol for the syndrome decoding problem.

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{u} \bmod 2,$$

for public $(\mathbf{A}, \mathbf{u})$ and secret binary vector $\mathbf{x}$ having fixed Hamming weight w.

Stern's Ideas

1. **Permuting:** Proving the witness constraint using random permutation.
   - Send the verifier $\pi(\mathbf{x})$.
   - $\mathbf{x}$ has constraint "binary vector with weight w" iff $\pi(\mathbf{x})$ does.

   The randomness of π protects the actual value of $\mathbf{x}$.
2. **Masking:** Proving the linear equation using a random masking $\mathbf{r}$.
   - Send the verifier $\mathbf{y} = \mathbf{x} + \mathbf{r}$, and show that: $\mathbf{A} \cdot \mathbf{y} = \mathbf{u} + \mathbf{A} \cdot \mathbf{r}$.

We will:

1. Pre-process the given "quadratic relation";
2. Exploit Stern's ideas, especially: permuting.

Dealing with Quadratic Relations: First Step

Goal

Transforming $\mathbf{X} \cdot \mathbf{s} =$ (public matrix) $\cdot$ (secret vector) mod q.

Dealing with Quadratic Relations: First Step

Goal

Transforming $\mathbf{X} \cdot \mathbf{s} = $ (public matrix) $\cdot$ (secret vector) mod q.

1. $\mathbf{X} \cdot \mathbf{s} = \sum_{i=1}^{n} \mathbf{x}_i \cdot s_i$, where $\mathbf{x}_i \in \mathbb{Z}_q^m$: columns of $\mathbf{X}$; and $s_i \in \mathbb{Z}_q$: entries of $\mathbf{s}$.

Dealing with Quadratic Relations: First Step

Goal

Transforming $\mathbf{X} \cdot \mathbf{s} = (\text{public matrix}) \cdot (\text{secret vector}) \bmod q$.

1. $\mathbf{X} \cdot \mathrm{s} = \sum_{i=1}^{n} \mathbf{x}_i \cdot s_i$, where $\mathbf{x}_i \in \mathbb{Z}_q^m$: columns of $\mathbf{X}$; and $s_i \in \mathbb{Z}_q$: entries of $\mathbf{s}$.
2. $\mathbf{x}_i \cdot s_i = \mathbf{H} \cdot \left(x_{i,1} \cdot s_i, \ldots x_{i,mk} \cdot s_i\right)^T$, where $k = \lceil \log_2 q \rceil$ and $\mathbf{H}$ is a public matrix allowing to decompose elements of $\mathbb{Z}_q$ into k bits.

Dealing with Quadratic Relations: First Step

Goal

Transforming $\mathbf{X} \cdot \mathbf{s} = (\text{public matrix}) \cdot (\text{secret vector}) \bmod q$.

1. $\mathbf{X} \cdot \mathbf{s} = \sum_{i=1}^{n} \mathbf{x}_i \cdot s_i$, where $\mathbf{x}_i \in \mathbb{Z}_q^m$: columns of $\mathbf{X}$; and $s_i \in \mathbb{Z}_q$: entries of $\mathbf{s}$.
2. $\mathbf{x}_i \cdot s_i = \mathbf{H} \cdot \big(x_{i,1} \cdot s_i, \ldots x_{i,mk} \cdot s_i\big)^T$, where $k = \lceil \log_2 q \rceil$ and $\mathbf{H}$ is a public matrix allowing to decompose elements of $\mathbb{Z}_q$ into k bits.
3. $x_{i,j} \cdot s_i = x_{i,j} \cdot (q_1, \ldots, q_k) \cdot (s_{i,1}, \ldots, s_{i,k})^T = (q_1, \ldots, q_k) \cdot (x_{i,j} \cdot s_{i,1}, \ldots, x_{i,j} \cdot s_{i,k})^T$.

Dealing with Quadratic Relations: First Step

Goal

Transforming $\mathbf{X} \cdot \mathbf{s} = \text{(public matrix)} \cdot \text{(secret vector)} \bmod q$.

1. $\mathbf{X} \cdot \mathrm{s} = \sum_{i=1}^{n} \mathbf{x}_i \cdot s_i$, where $\mathbf{x}_i \in \mathbb{Z}_q^m$: columns of $\mathbf{X}$; and $s_i \in \mathbb{Z}_q$: entries of $\mathbf{s}$.
2. $\mathbf{x}_i \cdot s_i = \mathbf{H} \cdot \left(x_{i,1} \cdot s_i, \ldots x_{i,mk} \cdot s_i\right)^T$, where $k = \lceil \log_2 q \rceil$ and $\mathbf{H}$ is a public matrix allowing to decompose elements of $\mathbb{Z}_q$ into k bits.
3. $x_{i,j} \cdot s_i = x_{i,j} \cdot (q_1, \ldots, q_k) \cdot (s_{i,1}, \ldots, s_{i,k})^T = (q_1, \ldots, q_k) \cdot (x_{i,j} \cdot s_{i,1}, \ldots, x_{i,j} \cdot s_{i,k})^T$.

$x_{i,j} \cdot s_i$ has form (public matrix)$\cdot$(secret vector) $\rightarrow$ so does $\mathbf{x}_i \cdot s_i$ $\rightarrow$ so does $\mathbf{X} \cdot \mathbf{s}$:

$$\mathbf{X} \cdot \mathbf{s} = \mathbf{Q} \cdot \mathbf{z} \bmod q,$$

where $\mathbf{Q} \in \mathbb{Z}_q^{m \times nmk^2}$ and $\mathbf{z} \in \{0,1\}^{nmk^2}$.

Dealing with Quadratic Relations: First Step

Goal

Transforming $\mathbf{X} \cdot \mathbf{s} =$ (public matrix) $\cdot$ (secret vector) mod q.

1. $\mathbf{X} \cdot \mathbf{s} = \sum_{i=1}^{n} \mathbf{x}_i \cdot s_i$, where $\mathbf{x}_i \in \mathbb{Z}_q^m$: columns of $\mathbf{X}$; and $s_i \in \mathbb{Z}_q$: entries of $\mathbf{s}$.
2. $\mathbf{x}_i \cdot s_i = \mathbf{H} \cdot \left(x_{i,1} \cdot s_i, \ldots x_{i,mk} \cdot s_i\right)^T$, where $k = \lceil \log_2 q \rceil$ and $\mathbf{H}$ is a public matrix allowing to decompose elements of $\mathbb{Z}_q$ into k bits.
3. $x_{i,j} \cdot s_i = x_{i,j} \cdot (q_1, \ldots, q_k) \cdot (s_{i,1}, \ldots, s_{i,k})^T = (q_1, \ldots, q_k) \cdot (x_{i,j} \cdot s_{i,1}, \ldots, x_{i,j} \cdot s_{i,k})^T$.

$x_{i,j} \cdot s_i$ has form (public matrix)$\cdot$(secret vector) $\rightarrow$ so does $\mathbf{x}_i \cdot s_i$ $\rightarrow$ so does $\mathbf{X} \cdot \mathbf{s}$:

$$\mathbf{X} \cdot \mathbf{s} = \mathbf{Q} \cdot \mathbf{z} \bmod q,$$

where $\mathbf{Q} \in \mathbb{Z}_q^{m \times nmk^2}$ and $\mathbf{z} \in \{0,1\}^{nmk^2}$.

- $\mathbf{z}$ is still "quadratic": each z_i is a product of a bit from $\mathbf{X}$ and a bit from $\mathbf{s}$.
- The component bits additionally satisfy other relations.

Dealing with Quadratic Relations: Second Step

A Divide-and-Conquer Strategy

Proving that a secret bit z has the form $z = c_1 \cdot c_2$, while preserving the possibility of showing that the component bits c_1 and c_2 satisfy other equations.

Dealing with Quadratic Relations: Second Step

A Divide-and-Conquer Strategy

Proving that a secret bit z has the form $z = c_1 \cdot c_2$, while preserving the possibility of showing that the component bits c_1 and c_2 satisfy other equations.

Technique: **Two-bit-based permuting**.

- For $c \in \{0, 1\}$, let $\overline{c} = 1 - c$. For $c_1, c_2 \in \{0, 1\}$, define the vector

$$\mathrm{ext}(c_1, c_2) = (\overline{c}_1 \cdot \overline{c}_2, \overline{c}_1 \cdot c_2, c_1 \cdot \overline{c}_2, c_1 \cdot c_2)^\top \in \{0, 1\}^4.$$

Dealing with Quadratic Relations: Second Step

A Divide-and-Conquer Strategy

Proving that a secret bit z has the form $z = c_1 \cdot c_2$, while preserving the possibility of showing that the component bits c_1 and c_2 satisfy other equations.

Technique: **Two-bit-based permuting**.

- For $c \in \{0,1\}$, let $\overline{c} = 1 - c$. For $c_1, c_2 \in \{0,1\}$, define the vector
$$\mathrm{ext}(c_1, c_2) = (\overline{c}_1 \cdot \overline{c}_2, \overline{c}_1 \cdot c_2, c_1 \cdot \overline{c}_2, c_1 \cdot c_2)^\top \in \{0,1\}^4.$$
- For $b_1, b_2 \in \{0,1\}$, define the permutation T_{b_1,b_2} that transforms vector
$$\mathbf{v} = (v_{0,0}, v_{0,1}, v_{1,0}, v_{1,1})^\top \in \mathbb{Z}^4$$
to vector $(v_{b_1,b_2}, v_{b_1,\overline{b}_2}, v_{\overline{b}_1,b_2}, v_{\overline{b}_1,\overline{b}_2})^\top$.

Dealing with Quadratic Relations: Second Step

A Divide-and-Conquer Strategy

Proving that a secret bit z has the form $z = c_1 \cdot c_2$, while preserving the possibility of showing that the component bits c_1 and c_2 satisfy other equations.

Technique: **Two-bit-based permuting**.

- For $c \in \{0,1\}$, let $\overline{c} = 1 - c$. For $c_1, c_2 \in \{0,1\}$, define the vector

$$\mathrm{ext}(c_1, c_2) = (\overline{c}_1 \cdot \overline{c}_2, \overline{c}_1 \cdot c_2, c_1 \cdot \overline{c}_2, c_1 \cdot c_2)^\top \in \{0,1\}^4.$$

- For $b_1, b_2 \in \{0,1\}$, define the permutation T_{b_1,b_2} that transforms vector

$$\mathbf{v} = (v_{0,0}, v_{0,1}, v_{1,0}, v_{1,1})^\top \in \mathbb{Z}^4$$

  to vector $(v_{b_1,b_2}, v_{b_1,\overline{b}_2}, v_{\overline{b}_1,b_2}, v_{\overline{b}_1,\overline{b}_2})^\top$.

  Note that, for all $c_1, c_2, b_1, b_2 \in \{0,1\}$, we have the equivalence:

$$\mathbf{v} = \mathrm{ext}(c_1, c_2) \iff T_{b_1,b_2}(\mathbf{v}) = \mathrm{ext}(c_1 \oplus b_1, c_2 \oplus b_2).$$

How Does It Work?

$$\mathbf{v} = \mathrm{ext}(c_1, c_2) \iff T_{b_1,b_2}(\mathbf{v}) = \mathrm{ext}(c_1 \oplus b_1, c_2 \oplus b_2).$$

Example: Let $c_1 = 1, c_2 = 0$. Then:

$$\begin{aligned} \mathbf{v} = \mathrm{ext}(c_1, c_2) &= (\bar{c}_1 \cdot \bar{c}_2, \bar{c}_1 \cdot c_2, c_1 \cdot \bar{c}_2, c_1 \cdot c_2)^\top \\ &= (0 \cdot 1, 0 \cdot 0, 1 \cdot 1, 1 \cdot 0)^T = (0, 0, 1, 0)^T. \end{aligned}$$

How Does It Work?

$$\mathbf{v} = \mathrm{ext}(c_1, c_2) \iff T_{b_1,b_2}(\mathbf{v}) = \mathrm{ext}(c_1 \oplus b_1, c_2 \oplus b_2).$$

Example: Let $c_1 = 1, c_2 = 0$. Then:

$$\begin{aligned} \mathbf{v} = \mathrm{ext}(c_1, c_2) &= (\bar{c}_1 \cdot \bar{c}_2, \bar{c}_1 \cdot c_2, c_1 \cdot \bar{c}_2, c_1 \cdot c_2)^\top \\ &= (0 \cdot 1, 0 \cdot 0, 1 \cdot 1, 1 \cdot 0)^T = (0, 0, 1, 0)^T. \end{aligned}$$

We have $v_{0,0} = 0, v_{0,1} = 0, v_{1,0} = 1, v_{1,1} = 0$. Now, let $b_1 = 1, b_2 = 1$.

$$\begin{aligned} T_{b_1,b_2}(\mathbf{v}) &= (v_{1,1}, v_{1,0}, v_{0,1}, v_{0,0})^\top = (0, 1, 0, 0)^T \\ &= \mathrm{ext}(0, 1) = \mathrm{ext}(1 \oplus 1, 0 \oplus 1) = \mathrm{ext}(c_1 \oplus b_1, c_2 \oplus b_2). \end{aligned}$$

How Does It Work?

$$\mathbf{v} = \mathrm{ext}(c_1, c_2) \iff T_{b_1,b_2}(\mathbf{v}) = \mathrm{ext}(c_1 \oplus b_1, c_2 \oplus b_2).$$

Example: Let $c_1 = 1, c_2 = 0$. Then:

$$\begin{aligned} \mathbf{v} = \mathrm{ext}(c_1, c_2) &= (\bar{c}_1 \cdot \bar{c}_2, \bar{c}_1 \cdot c_2, c_1 \cdot \bar{c}_2, c_1 \cdot c_2)^\top \\ &= (0 \cdot 1, 0 \cdot 0, 1 \cdot 1, 1 \cdot 0)^T = (0,0,1,0)^T. \end{aligned}$$

We have $v_{0,0} = 0, v_{0,1} = 0, v_{1,0} = 1, v_{1,1} = 0$. Now, let $b_1 = 1, b_2 = 1$.

$$\begin{aligned} T_{b_1,b_2}(\mathbf{v}) &= (v_{1,1}, v_{1,0}, v_{0,1}, v_{0,0})^\top = (0,1,0,0)^T \\ &= \mathrm{ext}(0,1) = \mathrm{ext}(1 \oplus 1, 0 \oplus 1) = \mathrm{ext}(c_1 \oplus b_1, c_2 \oplus b_2). \end{aligned}$$

Solution to the sub-problem:

1. Extend $z = c_1 \cdot c_2$ to $\mathbf{v} = \mathrm{ext}(c_1, c_2)$.
2. Permute $\mathbf{v}$ with random bits b_1, b_2, and give the verifier the permuted vector.
3. To prove that the **same** bits c_1, c_2 appear in other equations: set up similar mechanisms at their other appearances, and use the **same** b_1, b_2.

Putting Everything Together

- Our new Stern-like techniques allow to handle “quadratic relations”.

Putting Everything Together

- Our new Stern-like techniques allow to handle "quadratic relations".
- Ingredients for our GE instantiation:
    1. An anonymous CCA-secure PKE obtained from the [ABB - EC'10] IBE scheme, via the [CHK - EC'04] transformation.
    2. The signature scheme from [LLMNW - AC'16].

Putting Everything Together

- Our new Stern-like techniques allow to handle "quadratic relations".
- Ingredients for our GE instantiation:
  1. An anonymous CCA-secure PKE obtained from the [ABB - EC'10] IBE scheme, via the [CHK - EC'04] transformation.
  2. The signature scheme from [LLMNW - AC'16].
- Combining with known Stern-like techniques for encryption and signatures, we obtain the ZK protocol required for the GE.

Putting Everything Together

- Our new Stern-like techniques allow to handle "quadratic relations".
- Ingredients for our GE instantiation:
  1. An anonymous CCA-secure PKE obtained from the [ABB - EC'10] IBE scheme, via the [CHK - EC'04] transformation.
  2. The signature scheme from [LLMNW - AC'16].
- Combining with known Stern-like techniques for encryption and signatures, we obtain the ZK protocol required for the GE.

Thank you!