Invertible Polynomial Representation for Private Set Operations

Jung Hee Cheon, Hyunsook Hong, and Hyung Tae Lee

CHRI & Seoul National University

ICISC, November 29, 2013

Contents

Private Set Operations & Polynomial Representation

- Definition
  - Each participant $\mathcal{P}_i$ has own dataset $S_i \subset \mathcal{U}$.
  - (Informally,) all participants learn the resulting set operation among $S_1, \cdots, S_n$ without revealing any other information.

- Techniques for Private Set Operations
  - general MPC [GMW87, BGW87], pseudorandom function [JL09], blind RSA signature [DT10,DKT10]
  - polynomial representation [FNP04,KS05,Fri07,...]

- Polynomial representation of a set
  For a set $S = \{s_1, \ldots, s_k\}$, $f_S := \prod_{s_j \in S}(x - s_j) = \sum_{i=0}^{k} f_S[i]x^i$.

- Set operations from polynomial operations
  - Set intersection: $f_{S_1} + \ldots + f_{S_n} = \gcd(f_{S_1}, \ldots, f_{S_n}) \cdot u$
  - Multi-set Union: $f_1 \cdot f_2 \cdot \cdots \cdot f_n$

Private Set Operations & Polynomial Representation

- Definition
  - Each participant $\mathcal{P}_i$ has own dataset $S_i \subset \mathcal{U}$.
  - (Informally,) all participants learn the resulting set operation among $S_1, \cdots, S_n$ without revealing any other information.

- Techniques for Private Set Operations
  - general MPC [GMW87, BGW87], pseudorandom function [JL09], blind RSA signature [DT10,DKT10]
  - polynomial representation [FNP04,KS05,Fri07,...]

- Polynomial representation of a set
  For a set $S = \{s_1, \ldots, s_k\}$, $f_S := \prod_{s_j \in S}(x - s_j) = \sum_{i=0}^{k} f_S[i]x^i$.

- Set operations from polynomial operations
  - Set intersection: $f_{S_1} + \ldots + f_{S_n} = \gcd(f_{S_1}, \ldots, f_{S_n}) \cdot u$
  - Multi-set Union: $f_1 \cdot f_2 \cdot \cdots \cdot f_n$

Private Set Operations & Polynomial Representation

- Definition
  - Each participant $\mathcal{P}_i$ has own dataset $S_i \subset \mathcal{U}$.
  - (Informally,) all participants learn the resulting set operation among $S_1, \cdots, S_n$ without revealing any other information.

- Techniques for Private Set Operations
  - general MPC [GMW87, BGW87], pseudorandom function [JL09], blind RSA signature [DT10,DKT10]
  - polynomial representation [FNP04,KS05,Fri07,...]

- Polynomial representation of a set
  For a set $S = \{s_1, \ldots, s_k\}$, $f_S := \prod_{s_j \in S}(x - s_j) = \sum_{i=0}^{k} f_S[i]x^i$.

- Set operations from polynomial operations
  - Set intersection: $f_{S_1} + \ldots + f_{S_n} = \gcd(f_{S_1}, \ldots, f_{S_n}) \cdot u$
  - Multi-set Union: $f_1 \cdot f_2 \cdot \cdots \cdot f_n$

Private Set Operations & Polynomial Representation

- Definition
  - Each participant $\mathcal{P}_i$ has own dataset $S_i \subset \mathcal{U}$.
  - (Informally,) all participants learn the resulting set operation among $S_1, \cdots, S_n$ without revealing any other information.

- Techniques for Private Set Operations
  - general MPC [GMW87, BGW87], pseudorandom function [JL09], blind RSA signature [DT10,DKT10]
  - polynomial representation [FNP04,KS05,Fri07,...]

- Polynomial representation of a set
  For a set $S = \{s_1, \ldots, s_k\}$, $f_S := \prod_{s_j \in S}(x - s_j) = \sum_{i=0}^{k} f_S[i]x^i$.

- Set operations from polynomial operations
  - Set intersection: $f_{S_1} + \ldots + f_{S_n} = \gcd(f_{S_1}, \ldots, f_{S_n}) \cdot u$
  - Multi-set Union: $f_1 \cdot f_2 \cdot \cdots \cdot f_n$

Rational Function Representation

- Rational function representation of a set: For a set $S = \{s_1, \ldots, s_k\}$,

$$f_S := \frac{1}{\prod_{s_j \in S}(x - s_j)}.$$

- Set union from the sum of rational functions:

$$\frac{1}{f_{S_1}} + \ldots + \frac{1}{f_{S_n}} = \frac{u}{\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})}$$

- Seo et al. [SCK12] realized a rational function representation using reversed Laurent series and rational reconstruction algorithm.
  - Reversed Laurent series: Represent the inverse of polynomial of degree d as a polynomial of degree $2d + 1$
  - Rational reconstruction algorithm: Recover a denominator from a rational function

Rational Function Representation

- Rational function representation of a set: For a set $S = \{s_1, \ldots, s_k\}$,

$$f_S := \frac{1}{\prod_{s_j \in S}(x - s_j)}.$$

- Set union from the sum of rational functions:

$$\frac{1}{f_{S_1}} + \ldots + \frac{1}{f_{S_n}} = \frac{u}{\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})}$$

- Seo et al. [SCK12] realized a rational function representation using reversed Laurent series and rational reconstruction algorithm.
  - Reversed Laurent series: Represent the inverse of polynomial of degree d as a polynomial of degree $2d + 1$
  - Rational reconstruction algorithm: Recover a denominator from a rational function

Rational Function Representation

- Rational function representation of a set: For a set $S = \{s_1, \ldots, s_k\}$,

$$f_S := \frac{1}{\prod_{s_j \in S}(x - s_j)}.$$

- Set union from the sum of rational functions:

$$\frac{1}{f_{S_1}} + \ldots + \frac{1}{f_{S_n}} = \frac{u}{\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})}$$

- Seo et al. [SCK12] realized a rational function representation using reversed Laurent series and rational reconstruction algorithm.
  - Reversed Laurent series: Represent the inverse of polynomial of degree d as a polynomial of degree $2d + 1$
  - Rational reconstruction algorithm: Recover a denominator from a rational function

Polynomial Encryption & Operations

- Polynomial encryption: For a polynomial $f = \sum_{i=0}^{\deg f} f[i]x^i \in R[x]$, an encryption of the polynomial f is defined by

$$\mathcal{E}_{\mathsf{pk}}(f) := \sum_{i=0}^{\deg f} \mathsf{E}_{\mathsf{pk}}(f[i])x^i.$$

- If a utilized encryption E_{pk} has additive homomorphic property,
  - Polynomial addition: Given $\mathcal{E}_{\mathsf{pk}}(f), \mathcal{E}_{\mathsf{pk}}(g)$, compute $\mathcal{E}_{\mathsf{pk}}(f+g)$ by calculating
  $$\mathsf{E}_{\mathsf{pk}}((f+g)[i]) = \mathsf{E}_{\mathsf{pk}}(f[i])\mathsf{E}_{\mathsf{pk}}(g[i]).$$
  - Polynomial multiplication: Given $\mathcal{E}_{\mathsf{pk}}(f)$ and g, compute $\mathcal{E}_{\mathsf{pk}}(fg)$ by calculating
  $$\mathsf{E}_{\mathsf{pk}}((fg)[\ell]) = \prod_{i+j=\ell} \mathsf{E}_{\mathsf{pk}}(f[i])^{g[j]}.$$

Polynomial Encryption & Operations

- Polynomial encryption: For a polynomial $f = \sum_{i=0}^{\deg f} f[i]x^i \in R[x]$, an encryption of the polynomial f is defined by

$$\mathcal{E}_{\mathsf{pk}}(f) := \sum_{i=0}^{\deg f} \mathsf{E}_{\mathsf{pk}}(f[i])x^i.$$

- If a utilized encryption E_{pk} has additive homomorphic property,
  - Polynomial addition: Given $\mathcal{E}_{\mathsf{pk}}(f), \mathcal{E}_{\mathsf{pk}}(g)$, compute $\mathcal{E}_{\mathsf{pk}}(f+g)$ by calculating
  $$\mathsf{E}_{\mathsf{pk}}((f+g)[i]) = \mathsf{E}_{\mathsf{pk}}(f[i])\mathsf{E}_{\mathsf{pk}}(g[i]).$$
  - Polynomial multiplication: Given $\mathcal{E}_{\mathsf{pk}}(f)$ and g, compute $\mathcal{E}_{\mathsf{pk}}(fg)$ by calculating
  $$\mathsf{E}_{\mathsf{pk}}((fg)[\ell]) = \prod_{i+j=\ell} \mathsf{E}_{\mathsf{pk}}(f[i])^{g[j]}.$$

Application: Constant-round Set Union Protocol

Constant-round Private Set Union Protocol

1. Each player computes $\frac{1}{f_{S_i}}$ and sends an encryption of $\frac{1}{f_{S_i}}$.
2. All players jointly compute the encryption of a rational function

$$\frac{r_1}{f_{S_1}} + \ldots + \frac{r_n}{f_{S_n}} = \frac{u}{\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})}$$

   when r_i's are computed secretly by contributions of all players.
3. All players perform group decryption. Each player computes $\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})$ using rational reconstruction algorithm. Thereafter...?

Problem

How to recover the resulting set from $\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})$?

$\Rightarrow$ Find all roots of $\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})$!

Application: Constant-round Set Union Protocol

Constant-round Private Set Union Protocol

1. Each player computes $\frac{1}{f_{S_i}}$ and sends an encryption of $\frac{1}{f_{S_i}}$.
2. All players jointly compute the encryption of a rational function

$$\frac{r_1}{f_{S_1}} + \ldots + \frac{r_n}{f_{S_n}} = \frac{u}{\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})}$$

   when r_i's are computed secretly by contributions of all players.
3. All players perform group decryption. Each player computes $\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})$ using rational reconstruction algorithm. Thereafter...?

Problem

How to recover the resulting set from $\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})$?

$\Rightarrow$ Find all roots of $\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})$!

Application: Constant-round Set Union Protocol

Constant-round Private Set Union Protocol

1. Each player computes $\frac{1}{f_{S_i}}$ and sends an encryption of $\frac{1}{f_{S_i}}$.
2. All players jointly compute the encryption of a rational function

$$\frac{r_1}{f_{S_1}} + \ldots + \frac{r_n}{f_{S_n}} = \frac{u}{\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})}$$

   when r_i's are computed secretly by contributions of all players.
3. All players perform group decryption. Each player computes $\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})$ using rational reconstruction algorithm.
   Thereafter...?

Problem

How to recover the resulting set from $\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})$?

$\Rightarrow$ Find all roots of $\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})$!

Application: Constant-round Set Union Protocol

Constant-round Private Set Union Protocol

1. Each player computes $\frac{1}{f_{S_i}}$ and sends an encryption of $\frac{1}{f_{S_i}}$.
2. All players jointly compute the encryption of a rational function

$$\frac{r_1}{f_{S_1}} + \ldots + \frac{r_n}{f_{S_n}} = \frac{u}{\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})}$$

   when r_i's are computed secretly by contributions of all players.
3. All players perform group decryption. Each player computes $\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})$ using rational reconstruction algorithm.
   Thereafter...?

Problem

How to recover the resulting set from $\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})$?

$\Rightarrow$ Find all roots of $\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})$!

Application: Constant-round Set Union Protocol

Constant-round Private Set Union Protocol

1. Each player computes $\frac{1}{f_{S_i}}$ and sends an encryption of $\frac{1}{f_{S_i}}$.
2. All players jointly compute the encryption of a rational function

$$\frac{r_1}{f_{S_1}} + \ldots + \frac{r_n}{f_{S_n}} = \frac{u}{\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})}$$

   when r_i's are computed secretly by contributions of all players.
3. All players perform group decryption. Each player computes $\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})$ using rational reconstruction algorithm.
   Thereafter...?

Problem

How to recover the resulting set from $\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})$?

$\Rightarrow$ Find all roots of $\mathrm{lcm}(f_{S_1}, \ldots, f_{S_n})$!

Polynomial Factorization over Message Spaces

Table : Polynomial Factorization over Message Spaces

AHE	Message Space	Obstacle
Okamoto-Uchiyama	$\mathbb{Z}_p$	Hidden order
Paillier	$\mathbb{Z}_N$	$\Leftrightarrow$ Factoring N
Naccache-Stern	$\mathbb{Z}_\sigma$	$\mathbb{Z}_\sigma[x]$ is not UFD
Additive El Gamal	$\mathbb{Z}_p$	Heavy Decryption Cost
Some SWHE	$\mathbb{Z}_p$	Large PK Size, Ciphertext Size

- For example of Naccache-Stern case, in $\mathbb{Z}_6[x]$,

  $x^2-5x = x(x-5) = (x-2)(x-3) \Rightarrow$ Corresponded Set? $\{0,5\}$ or $\{2,3\}$

- Most previous protocols hired mix-net protocols.

  $\Rightarrow O(t)$ rounds for the number of malicious adversaries t!

- Seo et al. overcame this obstacle using secret sharing technique.

  $\Rightarrow$ Heavy computational cost and honest majority assumption!

Polynomial Factorization over Message Spaces

Table : Polynomial Factorization over Message Spaces

AHE	Message Space	Obstacle
Okamoto-Uchiyama	$\mathbb{Z}_p$	Hidden order
Paillier	$\mathbb{Z}_N$	$\Leftrightarrow$ Factoring N
Naccache-Stern	$\mathbb{Z}_\sigma$	$\mathbb{Z}_\sigma[x]$ is not UFD
Additive El Gamal	$\mathbb{Z}_p$	Heavy Decryption Cost
Some SWHE	$\mathbb{Z}_p$	Large PK Size, Ciphertext Size

- For example of Naccache-Stern case, in $\mathbb{Z}_6[x]$,

  $x^2-5x = x(x-5) = (x-2)(x-3) \Rightarrow$ Corresponded Set? $\{0,5\}$ or $\{2,3\}$

- Most previous protocols hired mix-net protocols.
  $\Rightarrow$ $O(t)$ rounds for the number of malicious adversaries t!
- Seo et al. overcame this obstacle using secret sharing technique.
  $\Rightarrow$ Heavy computational cost and honest majority assumption!

Polynomial Factorization over Message Spaces

Table : Polynomial Factorization over Message Spaces

AHE	Message Space	Obstacle
Okamoto-Uchiyama	$\mathbb{Z}_p$	Hidden order
Paillier	$\mathbb{Z}_N$	$\Leftrightarrow$ Factoring N
Naccache-Stern	$\mathbb{Z}_\sigma$	$\mathbb{Z}_\sigma[x]$ is not UFD
Additive El Gamal	$\mathbb{Z}_p$	Heavy Decryption Cost
Some SWHE	$\mathbb{Z}_p$	Large PK Size, Ciphertext Size

- For example of Naccache-Stern case, in $\mathbb{Z}_6[x]$,

  $x^2-5x = x(x-5) = (x-2)(x-3) \Rightarrow$ Corresponded Set? $\{0,5\}$ or $\{2,3\}$

- Most previous protocols hired mix-net protocols.
  $\Rightarrow$ $O(t)$ rounds for the number of malicious adversaries t!
- Seo et al. overcame this obstacle using secret sharing technique.
  $\Rightarrow$ Heavy computational cost and honest majority assumption!

Problem & Our Contributions

Problem

Can we give a polynomial representation that enables us to uniquely factorize in a polynomial ring defined over a message space of an additive homomorphic encryption?

Our Contributions

- Answer

  Yes, we provide a polynomial representation in $\mathbb{Z}_\sigma[x]$, where $\mathbb{Z}_\sigma$ is the message space of Naccache-Stern encryption.

- Application

  The first and efficient constant-round set union protocol without honest majority

Problem & Our Contributions

Problem

Can we give a polynomial representation that enables us to uniquely factorize in a polynomial ring defined over a message space of an additive homomorphic encryption?

Our Contributions

- Answer
  - Yes, we provide a polynomial representation in $\mathbb{Z}_\sigma[x]$, where $\mathbb{Z}_\sigma$ is the message space of Naccache-Stern encryption.
- Application
  - The first and efficient constant-round set union protocol without honest majority

Problem & Our Contributions

Problem

Can we give a polynomial representation that enables us to uniquely factorize in a polynomial ring defined over a message space of an additive homomorphic encryption?

Our Contributions

- Answer
  - Yes, we provide a polynomial representation in $\mathbb{Z}_\sigma[x]$, where $\mathbb{Z}_\sigma$ is the message space of Naccache-Stern encryption.
- Application
  - The first and efficient constant-round set union protocol without honest majority

Parameters of NS Encryption and Our Idea

- Parameters of Naccache-Stern encryption
  - P, Q: primes such that $P = 2p' \prod_{j=1}^{\bar{\ell}/2} q_j + 1, Q = 2q' \prod_{j=\bar{\ell}/2+1}^{\bar{\ell}} q_j + 1$ for large primes p', q' and small primes q_j's
  - N: a product of two primes P, Q
  - $\sigma = \prod_{j=1}^{\bar{\ell}} q_j$
  - $\mathbb{Z}_\sigma$: a message space of Naccache-Stern encryption

Our Idea

- Focus that the factorization of σ is public.
- Obtain all roots of f in $\mathbb{Z}_\sigma[x]$
  - find all roots of f in $\mathbb{Z}_{q_j}[x]$ (for all j)
  - apply Chinese Remainder Theorem
- It gives $d^{\bar{\ell}}$ candidates of roots.
- If we give some relation between roots in $\mathbb{Z}_{q_j}[x]$ and roots in $\mathbb{Z}_{q_{j+1}}[x]$, ...

Parameters of NS Encryption and Our Idea

- Parameters of Naccache-Stern encryption
  - P, Q: primes such that $P = 2p' \prod_{j=1}^{\bar{\ell}/2} q_j + 1, Q = 2q' \prod_{j=\bar{\ell}/2+1}^{\bar{\ell}} q_j + 1$ for large primes p', q' and small primes q_j's
  - N: a product of two primes P, Q
  - $\sigma = \prod_{j=1}^{\bar{\ell}} q_j$
  - $\mathbb{Z}_\sigma$: a message space of Naccache-Stern encryption

Our Idea

- Focus that the factorization of σ is public.
- Obtain all roots of f in $\mathbb{Z}_\sigma[x]$
  - find all roots of f in $\mathbb{Z}_{q_j}[x]$ for all j
  - apply Chinese Remainder Theorem
- It gives $d^{\bar{\ell}}$ candidates of roots.
- If we give some relation between roots in $\mathbb{Z}_{q_j}[x]$ and roots in $\mathbb{Z}_{q_{j+1}}[x]$, ...

Parameters of NS Encryption and Our Idea

- Parameters of Naccache-Stern encryption
  - P, Q: primes such that $P = 2p' \prod_{j=1}^{\bar{\ell}/2} q_j + 1, Q = 2q' \prod_{j=\bar{\ell}/2+1}^{\bar{\ell}} q_j + 1$ for large primes p', q' and small primes q_j's
  - N: a product of two primes P, Q
  - $\sigma = \prod_{j=1}^{\bar{\ell}} q_j$
  - $\mathbb{Z}_\sigma$: a message space of Naccache-Stern encryption

Our Idea

- Focus that the factorization of σ is public.
- Obtain all roots of f in $\mathbb{Z}_\sigma[x]$
  - find all roots of f in $\mathbb{Z}_{q_j}[x]$ for all j
  - apply Chinese Remainder Theorem
- It gives $d^{\bar{\ell}}$ candidates of roots.
- If we give some relation between roots in $\mathbb{Z}_{q_j}[x]$ and roots in $\mathbb{Z}_{q_{j+1}}[x]$, ...

Parameters of NS Encryption and Our Idea

- Parameters of Naccache-Stern encryption
  - P, Q: primes such that $P = 2p' \prod_{j=1}^{\bar{\ell}/2} q_j + 1, Q = 2q' \prod_{j=\bar{\ell}/2+1}^{\bar{\ell}} q_j + 1$ for large primes p', q' and small primes q_j's
  - N: a product of two primes P, Q
  - $\sigma = \prod_{j=1}^{\bar{\ell}} q_j$
  - $\mathbb{Z}_\sigma$: a message space of Naccache-Stern encryption

Our Idea

- Focus that the factorization of σ is public.
- Obtain all roots of f in $\mathbb{Z}_\sigma[x]$
  - find all roots of f in $\mathbb{Z}_{q_j}[x]$ for all j
  - apply Chinese Remainder Theorem
- It gives $d^{\bar{\ell}}$ candidates of roots.
- If we give some relation between roots in $\mathbb{Z}_{q_j}[x]$ and roots in $\mathbb{Z}_{q_{j+1}}[x]$, ...

Parameters of NS Encryption and Our Idea

- Parameters of Naccache-Stern encryption
  - P, Q: primes such that $P = 2p' \prod_{j=1}^{\bar{\ell}/2} q_j + 1, Q = 2q' \prod_{j=\bar{\ell}/2+1}^{\bar{\ell}} q_j + 1$ for large primes p', q' and small primes q_j's
  - N: a product of two primes P, Q
  - $\sigma = \prod_{j=1}^{\bar{\ell}} q_j$
  - $\mathbb{Z}_\sigma$: a message space of Naccache-Stern encryption

Our Idea

- Focus that the factorization of σ is public.
- Obtain all roots of f in $\mathbb{Z}_\sigma[x]$
  - find all roots of f in $\mathbb{Z}_{q_j}[x]$ for all j
  - apply Chinese Remainder Theorem
- It gives $d^{\bar{\ell}}$ candidates of roots.
- If we give some relation between roots in $\mathbb{Z}_{q_j}[x]$ and roots in $\mathbb{Z}_{q_{j+1}}[x]$, ...

Parameters of NS Encryption and Our Idea

- Parameters of Naccache-Stern encryption
  - P, Q: primes such that $P = 2p' \prod_{j=1}^{\bar{\ell}/2} q_j + 1, Q = 2q' \prod_{j=\bar{\ell}/2+1}^{\bar{\ell}} q_j + 1$ for large primes p', q' and small primes q_j's
  - N: a product of two primes P, Q
  - $\sigma = \prod_{j=1}^{\bar{\ell}} q_j$
  - $\mathbb{Z}_\sigma$: a message space of Naccache-Stern encryption

Our Idea

- Focus that the factorization of σ is public.
- Obtain all roots of f in $\mathbb{Z}_\sigma[x]$
  - find all roots of f in $\mathbb{Z}_{q_j}[x]$ for all j
  - apply Chinese Remainder Theorem
- It gives $d^{\bar{\ell}}$ candidates of roots.
- If we give some relation between roots in $\mathbb{Z}_{q_j}[x]$ and roots in $\mathbb{Z}_{q_{j+1}}[x]$, ...

Parameters of NS Encryption and Our Idea

- Parameters of Naccache-Stern encryption
  - P, Q: primes such that $P = 2p' \prod_{j=1}^{\bar{\ell}/2} q_j + 1$, $Q = 2q' \prod_{j=\bar{\ell}/2+1}^{\bar{\ell}} q_j + 1$ for large primes p', q' and small primes q_j's
  - N: a product of two primes P, Q
  - $\sigma = \prod_{j=1}^{\bar{\ell}} q_j$
  - $\mathbb{Z}_\sigma$: a message space of Naccache-Stern encryption

Our Idea

- Focus that the factorization of σ is public.
- Obtain all roots of f in $\mathbb{Z}_\sigma[x]$
  - find all roots of f in $\mathbb{Z}_{q_j}[x]$ for all j
  - apply Chinese Remainder Theorem
- It gives $d^{\bar{\ell}}$ candidates of roots.
- If we give some relation between roots in $\mathbb{Z}_{q_j}[x]$ and roots in $\mathbb{Z}_{q_{j+1}}[x]$, ...

Linkable Pair

- $\sigma = \prod_{i=1}^{\bar{\ell}} q_j$ and q_j's are $(3\tau + 1)$-bit primes.
- Parse $s_i \in \mathcal{U} \subset \{0,1\}^{\tau\ell}$ into ℓ τ-bit strings $s_{i,j}$'s.

Linkable Pair

1. $(s_j^{(i)}, s_{j+1}^{(i')}) \in \mathbb{Z}_{q_j} \times \mathbb{Z}_{q_{j+1}}$ is a linkable pair of length 2 if $s_{i,j+1}||s_{i,j+2} = s_{i',j+1}||s_{i',j+2}$.
2. We also define $(s_1^{(i_1)}, \cdots, s_{j+1}^{(i_{j+1})}) \in \mathbb{Z}_{q_1} \times \cdots \times \mathbb{Z}_{q_{j+1}}$ to be a *linkable pair* of length $j+1$ if $(s_1^{(i_1)}, \cdots, s_j^{(i_j)})$ and $(s_j^{(i_j)}, s_{j+1}^{(i_{j+1})})$ are linkable pairs of length j and 2, respectively.

- Example

$$
\begin{array}{lcccccccccl}
s_1^{(i_1)} & = & s_{i_1,1} & || & s_{i_1,2} & || & s_{i_1,3} & & & & \bmod q_1 \\
s_2^{(i_2)} & = & & & s_{i_2,2} & || & s_{i_2,3} & || & s_{i_2,4} & & \bmod q_2 \\
s_3^{(i_3)} & = & & & & & s_{i_3,3} & || & s_{i_3,4} & || \; s_{i_3,5} & \bmod q_3
\end{array}
$$

$\Rightarrow$ Then $(s_1^{(i_1)}, s_2^{(i_2)}, s_3^{(i_3)})$ is a linkable pair of length 3.

Encoding Phase

- $h : \{0,1\}^{\tau\ell} \to \{0,1\}^{2\tau}, h_j : \{0,1\}^{\tau\ell} \to \{0,1\}^{\tau}$: uniform hash functions for $1 \le j \le \ell'$.

1. Parse s_i into ℓ τ-bit strings $s_{i,j}$'s, *i.e.*, $s_i = s_{i,1}||s_{i,2}||s_{i,3}||\cdots||s_{i,\ell}$.
2. Set $s_{i,\ell+j} = h_j(s_i)$ for $1 \le j \le \ell'$ and $s_{i,\bar{\ell}+1}||s_{i,\bar{\ell}+2} = h(s_i)$.
3. Compute an encoding function $\iota : \mathcal{U} \subseteq \{0,1\}^{\tau\ell} \to \mathbb{Z}_\sigma$ so that $(\iota(s_i) \mod q_1, \iota(s_i) \mod q_2, \cdots, \iota(s_i) \mod q_{\bar{\ell}})$ is a linkable pair of length $\bar{\ell}$, *i.e.*,

$$\begin{aligned}
\iota(s_i) \mod q_1 &= s_{i,1} || s_{i,2} || s_{i,3} \\
\mod q_2 &= s_{i,2} || s_{i,3} || s_{i,4} \\
&\vdots \\
\mod q_{\bar{\ell}-1} &= s_{i,\bar{\ell}-1} || s_{i,\bar{\ell}} || s_{i,\bar{\ell}+1} \\
\mod q_{\bar{\ell}} &= s_{i,\bar{\ell}} || s_{i,\bar{\ell}+1} || s_{i,\bar{\ell}+2}.
\end{aligned}$$

4. Compute $f_S = \prod_{s_j \in S}(x - \iota(s_j)) \in \mathbb{Z}_\sigma[x]$.

- Complexity: $d(\ell'+1)$ hash computations + $O(d)$ CRT computations + $\tilde{O}(d)$ multiplications $\le O(d^2)$ exponentiations

Encoding Phase

- $h : \{0,1\}^{\tau\ell} \to \{0,1\}^{2\tau}, h_j : \{0,1\}^{\tau\ell} \to \{0,1\}^{\tau}$: uniform hash functions for $1 \le j \le \ell'$.

1. Parse s_i into ℓ τ-bit strings $s_{i,j}$'s, *i.e.*, $s_i = s_{i,1}||s_{i,2}||s_{i,3}||\cdots||s_{i,\ell}$.
2. Set $s_{i,\ell+j} = h_j(s_i)$ for $1 \le j \le \ell'$ and $s_{i,\bar{\ell}+1}||s_{i,\bar{\ell}+2} = h(s_i)$.
3. Compute an encoding function $\iota : \mathcal{U} \subseteq \{0,1\}^{\tau\ell} \to \mathbb{Z}_\sigma$ so that $(\iota(s_i) \mod q_1, \iota(s_i) \mod q_2, \cdots, \iota(s_i) \mod q_{\bar{\ell}})$ is a linkable pair of length $\bar{\ell}$, *i.e.*,

$$\begin{aligned}
\iota(s_i) \mod q_1 &= s_{i,1} \,||\, s_{i,2} \,||\, s_{i,3} \\
\mod q_2 &= s_{i,2} \,||\, s_{i,3} \,||\, s_{i,4} \\
&\vdots \\
\mod q_{\bar{\ell}-1} &= s_{i,\bar{\ell}-1} \,||\, s_{i,\bar{\ell}} \,||\, s_{i,\bar{\ell}+1} \\
\mod q_{\bar{\ell}} &= s_{i,\bar{\ell}} \,||\, s_{i,\bar{\ell}+1} \,||\, s_{i,\bar{\ell}+2}.
\end{aligned}$$

4. Compute $f_S = \prod_{s_j \in S}(x - \iota(s_j)) \in \mathbb{Z}_\sigma[x]$.

- Complexity: $d(\ell'+1)$ hash computations + $O(d)$ CRT computations + $\tilde{O}(d)$ multiplications $\le O(d^2)$ exponentiations

Encoding Phase

- $h : \{0,1\}^{\tau\ell} \to \{0,1\}^{2\tau}, h_j : \{0,1\}^{\tau\ell} \to \{0,1\}^{\tau}$: uniform hash functions for $1 \le j \le \ell'$.

1. Parse s_i into ℓ τ-bit strings $s_{i,j}$'s, *i.e.*, $s_i = s_{i,1}||s_{i,2}||s_{i,3}||\cdots||s_{i,\ell}$.
2. Set $s_{i,\ell+j} = h_j(s_i)$ for $1 \le j \le \ell'$ and $s_{i,\bar{\ell}+1}||s_{i,\bar{\ell}+2} = h(s_i)$.
3. Compute an encoding function $\iota : \mathcal{U} \subseteq \{0,1\}^{\tau\ell} \to \mathbb{Z}_\sigma$ so that $(\iota(s_i) \mod q_1, \iota(s_i) \mod q_2, \cdots, \iota(s_i) \mod q_{\bar{\ell}})$ is a linkable pair of length $\bar{\ell}$, *i.e.*,

$$\begin{aligned}
\iota(s_i) \mod q_1 &= s_{i,1} || s_{i,2} || s_{i,3} \\
\mod q_2 &= s_{i,2} || s_{i,3} || s_{i,4} \\
&\vdots \\
\mod q_{\bar{\ell}-1} &= s_{i,\bar{\ell}-1} || s_{i,\bar{\ell}} || s_{i,\bar{\ell}+1} \\
\mod q_{\bar{\ell}} &= s_{i,\bar{\ell}} || s_{i,\bar{\ell}+1} || s_{i,\bar{\ell}+2}.
\end{aligned}$$

4. Compute $f_S = \prod_{s_j \in S}(x - \iota(s_j)) \in \mathbb{Z}_\sigma[x]$.

- Complexity: $d(\ell'+1)$ hash computations + $O(d)$ CRT computations + $\tilde{O}(d)$ multiplications $\le O(d^2)$ exponentiations

Encoding Phase

- $h : \{0,1\}^{\tau\ell} \to \{0,1\}^{2\tau}, h_j : \{0,1\}^{\tau\ell} \to \{0,1\}^{\tau}$: uniform hash functions for $1 \le j \le \ell'$.

1. Parse s_i into ℓ τ-bit strings $s_{i,j}$'s, *i.e.*, $s_i = s_{i,1}||s_{i,2}||s_{i,3}||\cdots||s_{i,\ell}$.
2. Set $s_{i,\ell+j} = h_j(s_i)$ for $1 \le j \le \ell'$ and $s_{i,\bar{\ell}+1}||s_{i,\bar{\ell}+2} = h(s_i)$.
3. Compute an encoding function $\iota : \mathcal{U} \subseteq \{0,1\}^{\tau\ell} \to \mathbb{Z}_\sigma$ so that $(\iota(s_i) \mod q_1, \iota(s_i) \mod q_2, \cdots, \iota(s_i) \mod q_{\bar{\ell}})$ is a linkable pair of length $\bar{\ell}$, *i.e.*,

$$\begin{aligned}
\iota(s_i) \mod q_1 &= s_{i,1} || s_{i,2} || s_{i,3} \\
\mod q_2 &= s_{i,2} || s_{i,3} || s_{i,4} \\
&\vdots \\
\mod q_{\bar{\ell}-1} &= s_{i,\bar{\ell}-1} || s_{i,\bar{\ell}} || s_{i,\bar{\ell}+1} \\
\mod q_{\bar{\ell}} &= s_{i,\bar{\ell}} || s_{i,\bar{\ell}+1} || s_{i,\bar{\ell}+2}.
\end{aligned}$$

4. Compute $f_S = \prod_{s_j \in S}(x - \iota(s_j)) \in \mathbb{Z}_\sigma[x]$.

- Complexity: $d(\ell'+1)$ hash computations + $O(d)$ CRT computations + $\tilde{O}(d)$ multiplications $\le O(d^2)$ exponentiations

Encoding Phase

- $h : \{0,1\}^{\tau\ell} \to \{0,1\}^{2\tau}, h_j : \{0,1\}^{\tau\ell} \to \{0,1\}^{\tau}$: uniform hash functions for $1 \le j \le \ell'$.

1. Parse s_i into ℓ τ-bit strings $s_{i,j}$'s, *i.e.*, $s_i = s_{i,1}||s_{i,2}||s_{i,3}||\cdots||s_{i,\ell}$.
2. Set $s_{i,\ell+j} = h_j(s_i)$ for $1 \le j \le \ell'$ and $s_{i,\bar{\ell}+1}||s_{i,\bar{\ell}+2} = h(s_i)$.
3. Compute an encoding function $\iota : \mathcal{U} \subseteq \{0,1\}^{\tau\ell} \to \mathbb{Z}_\sigma$ so that $(\iota(s_i) \mod q_1, \iota(s_i) \mod q_2, \cdots, \iota(s_i) \mod q_{\bar{\ell}})$ is a linkable pair of length $\bar{\ell}$, *i.e.*,

$$\begin{aligned}
\iota(s_i) \mod q_1 &= s_{i,1} || s_{i,2} || s_{i,3} \\
\mod q_2 &= s_{i,2} || s_{i,3} || s_{i,4} \\
&\vdots \\
\mod q_{\bar{\ell}-1} &= s_{i,\bar{\ell}-1} || s_{i,\bar{\ell}} || s_{i,\bar{\ell}+1} \\
\mod q_{\bar{\ell}} &= s_{i,\bar{\ell}} || s_{i,\bar{\ell}+1} || s_{i,\bar{\ell}+2}.
\end{aligned}$$

4. Compute $f_S = \prod_{s_j \in S}(x - \iota(s_j)) \in \mathbb{Z}_\sigma[x]$.

- Complexity: $d(\ell'+1)$ hash computations + $O(d)$ CRT computations + $\tilde{O}(d)$ multiplications $\le O(d^2)$ exponentiations

Encoding Phase

- $h : \{0,1\}^{\tau\ell} \to \{0,1\}^{2\tau}, h_j : \{0,1\}^{\tau\ell} \to \{0,1\}^{\tau}$: uniform hash functions for $1 \le j \le \ell'$.

1. Parse s_i into ℓ τ-bit strings $s_{i,j}$'s, *i.e.*, $s_i = s_{i,1}||s_{i,2}||s_{i,3}||\cdots||s_{i,\ell}$.
2. Set $s_{i,\ell+j} = h_j(s_i)$ for $1 \le j \le \ell'$ and $s_{i,\bar{\ell}+1}||s_{i,\bar{\ell}+2} = h(s_i)$.
3. Compute an encoding function $\iota : \mathcal{U} \subseteq \{0,1\}^{\tau\ell} \to \mathbb{Z}_\sigma$ so that $(\iota(s_i) \mod q_1, \iota(s_i) \mod q_2, \cdots, \iota(s_i) \mod q_{\bar{\ell}})$ is a linkable pair of length $\bar{\ell}$, *i.e.*,

$$\begin{aligned}
\iota(s_i) \mod q_1 &= s_{i,1} || s_{i,2} || s_{i,3} \\
\mod q_2 &= s_{i,2} || s_{i,3} || s_{i,4} \\
&\vdots \\
\mod q_{\bar{\ell}-1} &= s_{i,\bar{\ell}-1} || s_{i,\bar{\ell}} || s_{i,\bar{\ell}+1} \\
\mod q_{\bar{\ell}} &= s_{i,\bar{\ell}} || s_{i,\bar{\ell}+1} || s_{i,\bar{\ell}+2}.
\end{aligned}$$

4. Compute $f_S = \prod_{s_j \in S}(x - \iota(s_j)) \in \mathbb{Z}_\sigma[x]$.

- Complexity: $d(\ell'+1)$ hash computations + $O(d)$ CRT computations + $\tilde{O}(d)$ multiplications $\le O(d^2)$ exponentiations

Analysis

- All roots of f_S are to be linkable pairs of length $\bar{\ell}$.

Theorem

- ι: our encoding function from $\{0,1\}^{\tau\ell}$ to $\mathbb{Z}_\sigma$
- h, h_j: uniform hash functions utilized in our encoding function ι

$\Rightarrow$ The expected number of linkable pairs of length $\bar{\ell}$ is at most $3d$ for all polynomials $f_S = \prod_{s_i \in S}(x - \iota(s_j))$ of degree d.

- Remove candidates which are linkable pairs, but not roots by checking hash values h and h_j's.
- The proper number of h_j's: $\ell' > \frac{3(\lambda + \log d)}{\log d + 2\log d_0} - 2$
  $\Rightarrow$ The probability that irrelevant candidates of roots are not filtered is negligible in the security parameter λ.

Analysis

- All roots of f_S are to be linkable pairs of length $\bar{\ell}$.

Theorem

- ι: our encoding function from $\{0,1\}^{\tau\ell}$ to $\mathbb{Z}_\sigma$
- h, h_j: uniform hash functions utilized in our encoding function ι

$\Rightarrow$ The expected number of linkable pairs of length $\bar{\ell}$ is at most $3d$ for all polynomials $f_S = \prod_{s_i \in S}(x - \iota(s_j))$ of degree d.

- Remove candidates which are linkable pairs, but not roots by checking hash values h and h_j's.
- The proper number of h_j's: $\ell' > \frac{3(\lambda + \log d)}{\log d + 2\log d_0} - 2$
  $\Rightarrow$ The probability that irrelevant candidates of roots are not filtered is negligible in the security parameter λ.

Analysis

- All roots of f_S are to be linkable pairs of length $\bar{\ell}$.

Theorem

- ι: our encoding function from $\{0,1\}^{\tau\ell}$ to $\mathbb{Z}_\sigma$
- h, h_j: uniform hash functions utilized in our encoding function ι

$\Rightarrow$ The expected number of linkable pairs of length $\bar{\ell}$ is at most $3d$ for all polynomials $f_S = \prod_{s_i \in S}(x - \iota(s_j))$ of degree d.

- Remove candidates which are linkable pairs, but not roots by checking hash values h and h_j's.
- The proper number of h_j's: $\ell' > \frac{3(\lambda + \log d)}{\log d + 2\log d_0} - 2$
  $\Rightarrow$ The probability that irrelevant candidates of roots are not filtered is negligible in the security parameter λ.

Analysis

- All roots of f_S are to be linkable pairs of length $\bar{\ell}$.

Theorem

- ι: our encoding function from $\{0,1\}^{\tau\ell}$ to $\mathbb{Z}_\sigma$
- h, h_j: uniform hash functions utilized in our encoding function ι

$\Rightarrow$ The expected number of linkable pairs of length $\bar{\ell}$ is at most $3d$ for all polynomials $f_S = \prod_{s_i \in S}(x - \iota(s_j))$ of degree d.

- Remove candidates which are linkable pairs, but not roots by checking hash values h and h_j's.
- The proper number of h_j's: $\ell' > \frac{3(\lambda + \log d)}{\log d + 2\log d_0} - 2$
  $\Rightarrow$ The probability that irrelevant candidates of roots are not filtered is negligible in the security parameter λ.

Decoding Phase

For a given polynomial $f(x) = \prod_{i=1}^{d}(x - \iota(s_i)) \in \mathbb{Z}_\sigma[x]$,

1. compute all roots $\{s_j^{(1)}, \cdots, s_j^{(d)}\}$ in $\mathbb{Z}_{q_j}[x]$ for each j,
2. find all the linkable pairs among $\{s_j^{(1)}, \cdots, s_j^{(d)}\}$ and $\{s_{j+1}^{(1)}, \cdots, s_{j+1}^{(d)}\}$ for each j sequentially from 1 to $\bar{\ell} - 1$,
3. check hash values $h(s_i)$ and $h_j(s_i)$ of the linkable pair s_i.

- Complexity:

$$
\begin{aligned}
& (O(\bar{\ell} d^{1.5+o(1)} \log^2 q) + O(\bar{\ell} d \log d \log q)) \text{ bit operations} \\
+ \quad & 3d\ell' \text{ hash computations} \\
\leq \quad & O(d^2) \text{ exponentiations}
\end{aligned}
$$

Decoding Phase

For a given polynomial $f(x) = \prod_{i=1}^{d}(x - \iota(s_i)) \in \mathbb{Z}_\sigma[x]$,

1. compute all roots $\{s_j^{(1)}, \cdots, s_j^{(d)}\}$ in $\mathbb{Z}_{q_j}[x]$ for each j,
2. find all the linkable pairs among $\{s_j^{(1)}, \cdots, s_j^{(d)}\}$ and $\{s_{j+1}^{(1)}, \cdots, s_{j+1}^{(d)}\}$ for each j sequentially from 1 to $\bar{\ell} - 1$,
3. check hash values $h(s_i)$ and $h_j(s_i)$ of the linkable pair s_i.

- Complexity:

$$
\begin{aligned}
& (O(\bar{\ell} d^{1.5+o(1)} \log^2 q) + O(\bar{\ell} d \log d \log q)) \text{ bit operations} \\
+ \quad & 3 d \ell' \text{ hash computations} \\
\leq \quad & O(d^2) \text{ exponentiations}
\end{aligned}
$$

Decoding Phase

For a given polynomial $f(x) = \prod_{i=1}^{d}(x - \iota(s_i)) \in \mathbb{Z}_\sigma[x]$,

1. compute all roots $\{s_j^{(1)}, \cdots, s_j^{(d)}\}$ in $\mathbb{Z}_{q_j}[x]$ for each j,
2. find all the linkable pairs among $\{s_j^{(1)}, \cdots, s_j^{(d)}\}$ and $\{s_{j+1}^{(1)}, \cdots, s_{j+1}^{(d)}\}$ for each j sequentially from 1 to $\bar{\ell} - 1$,
3. check hash values $h(s_i)$ and $h_j(s_i)$ of the linkable pair s_i.

- Complexity:

$$
\begin{aligned}
& (O(\bar{\ell} d^{1.5+o(1)} \log^2 q) + O(\bar{\ell} d \log d \log q)) \text{ bit operations} \\
+ \quad & 3d\ell' \text{ hash computations} \\
\leq \quad & O(d^2) \text{ exponentiations}
\end{aligned}
$$

Decoding Phase

For a given polynomial $f(x) = \prod_{i=1}^{d}(x - \iota(s_i)) \in \mathbb{Z}_\sigma[x]$,

1. compute all roots $\{s_j^{(1)}, \cdots, s_j^{(d)}\}$ in $\mathbb{Z}_{q_j}[x]$ for each j,
2. find all the linkable pairs among $\{s_j^{(1)}, \cdots, s_j^{(d)}\}$ and $\{s_{j+1}^{(1)}, \cdots, s_{j+1}^{(d)}\}$ for each j sequentially from 1 to $\bar{\ell} - 1$,
3. check hash values $h(s_i)$ and $h_j(s_i)$ of the linkable pair s_i.

- Complexity:

$$
\begin{aligned}
& (O(\bar{\ell} d^{1.5+o(1)} \log^2 q) + O(\bar{\ell} d \log d \log q)) \text{ bit operations} \\
+ \quad & 3d\ell' \text{ hash computations} \\
\leq \quad & O(d^2) \text{ exponentiations}
\end{aligned}
$$

Efficiency Comparison

Table : Comparison with Previous Set-Union Protocols

HBC	Rounds	Communication Cost	Computational Cost	# of Honest Party
[KS05]	$O(n)$	$O(n^3 k\tau_N)$	$O(n^4 k^2 \tau_N \rho_N)$	≥ 1
[Fri07]	$O(n)$	$O(n^2 k\tau_N)$	$O(n^2 k^2 \tau_N \rho_N)$	≥ 1
[SCK12]	$O(1)$	$O(n^4 k^2 \tau_{p'})$	$O(n^5 k^2 \rho_{p'})$	$\geq n/2$
Ours	$O(1)$	$O(n^3 k\tau_N)$	$O(n^3 k^2 \tau_N \rho_N)$	≥ 1
Malicious	**Rounds**	**Communication Cost**	**Computational Cost**	**# of Honest Party**
[Fri07]	$O(n)$	$O((n^2 k^2 + n^3 k)\tau_N)$	$O(n^2 k^2 \tau_N \rho_N)$	≥ 1
[SCK12]	$O(1)$	$O(n^4 k^2 \tau_p)$	$O(n^5 k^2 \tau_p \rho_p)$	$\geq n/2$
Ours	$O(1)$	$O(n^3 k^2 \tau_N)$	$O(n^3 k^2 \tau_N \rho_N)$	≥ 1

n: the number of participants, k: the maximum size of sets

$\tau_N, \tau_{p'}, \tau_p$: the size of modulus N for Paillier encryption scheme or NS encryption scheme, the size p' of representing domain, the order p of a cyclic group for Pedersen commitment scheme, respectively

$\rho_N, \rho_{p'}, \rho_p$: modular multiplication cost of modulus N for Paillier encryption scheme or NS encryption scheme, p' for the size of representing domain, p for the order of a cyclic group for Pedersen commitment scheme, respectively

Conclusion

Problem

Can we give a polynomial representation that enables us to uniquely factorize in a polynomial ring defined over a message space of an additive homomorphic encryption?

Our Contributions

- Answer

  Yes, we provide a polynomial representation in $\mathbb{Z}_\sigma[x]$, where $\mathbb{Z}_\sigma$ is the message space of Naccache-Stern encryption.

- Application

  The first and efficient constant-round set union protocol without honest majority

Conclusion

Problem

Can we give a polynomial representation that enables us to uniquely factorize in a polynomial ring defined over a message space of an additive homomorphic encryption?

Our Contributions

- Answer
  - Yes, we provide a polynomial representation in $\mathbb{Z}_\sigma[x]$, where $\mathbb{Z}_\sigma$ is the message space of Naccache-Stern encryption.
- Application
  - The first and efficient constant-round set union protocol without honest majority

Conclusion

Problem

Can we give a polynomial representation that enables us to uniquely factorize in a polynomial ring defined over a message space of an additive homomorphic encryption?

Our Contributions

- Answer
  - Yes, we provide a polynomial representation in $\mathbb{Z}_\sigma[x]$, where $\mathbb{Z}_\sigma$ is the message space of Naccache-Stern encryption.
- Application
  - The first and efficient constant-round set union protocol without honest majority

Conclusion

Problem

Can we give a polynomial representation that enables us to uniquely factorize in a polynomial ring defined over a message space of an additive homomorphic encryption?

Our Contributions

- Answer
  - Yes, we provide a polynomial representation in $\mathbb{Z}_\sigma[x]$, where $\mathbb{Z}_\sigma$ is the message space of Naccache-Stern encryption.
- Application
  - The first and efficient constant-round set union protocol without honest majority

********** THANK YOU !!!**********