A New Additive Homomorphic Encryption based on the co-ACD Problem

Jung Hee Cheon[1], Hyung Tae Lee[2], and Jae Hong Seo[3]

[1]Seoul National University, Korea

[2]Nanyang Technological University, Singapore

[3]Myongji University, Korea

ACM CCS 2014, November 04, 2014

Applications of Additive Homomorphic Encryption

- Basic applications: Statistics as encrypted
  - Computing average on encrypted data
- Advanced applications: Before the appearance of FHE, AHE enables us to construct various applications.
  - Oblivious pseudorandom functions, Oblivious transfer
  - Private information retrieval, Private set operation protocols
  - Electronic voting, Commitment scheme and so on
- Still, AHE-based applications are more efficient than FHE-based applications.

Applications of Additive Homomorphic Encryption

- Basic applications: Statistics as encrypted
  - Computing average on encrypted data
- Advanced applications: Before the appearance of FHE, AHE enables us to construct various applications.
  - Oblivious pseudorandom functions, Oblivious transfer
  - Private information retrieval, Private set operation protocols
  - Electronic voting, Commitment scheme and so on
- Still, AHE-based applications are more efficient than FHE-based applications.

Applications of Additive Homomorphic Encryption

- Basic applications: Statistics as encrypted
  - Computing average on encrypted data
- Advanced applications: Before the appearance of FHE, AHE enables us to construct various applications.
  - Oblivious pseudorandom functions, Oblivious transfer
  - Private information retrieval, Private set operation protocols
  - Electronic voting, Commitment scheme and so on
- Still, AHE-based applications are more efficient than FHE-based applications.

Our Results

- Strategy: Follow the technique to construct the recent SHE
  1. Construct secure private-key AHE
     - ★ Using modular reduction with several moduli and inserting Noise
     - ★ Analyze the hardness of a new problem by applying known attacks
  2. Convert into a public-key version
     - ★ $M + \sum \mathrm{Enc}(0)$ and leftover hash lemma over lattices

- Implementation result (128-bit security)

	Ctxt	PK	KeyGen	Enc	Dec	Add
Paillier	6144 bit	1.5 KB	437.39 s	62.46 ms	40.38 ms	12.40 μs
Ours	3072 bit	1.3 MB	0.35 s	0.72 ms	4.00 μs	0.40 μs

- Provide applications of our construction and general AHE

Our Results

- Strategy: Follow the technique to construct the recent SHE
  1. Construct secure private-key AHE
     - ★ Using modular reduction with several moduli and inserting Noise
     - ★ Analyze the hardness of a new problem by applying known attacks
  2. Convert into a public-key version
     - ★ $M + \sum \mathsf{Enc}(0)$ and leftover hash lemma over lattices

- Implementation result (128-bit security)

	Ctxt	PK	KeyGen	Enc	Dec	Add
Paillier	6144 bit	1.5 KB	437.39 s	62.46 ms	40.38 ms	12.40 μs
Ours	3072 bit	1.3 MB	0.35 s	0.72 ms	4.00 μs	0.40 μs

- Provide applications of our construction and general AHE

Our Results

- Strategy: Follow the technique to construct the recent SHE
  1. Construct secure private-key AHE
     - ★ Using modular reduction with several moduli and inserting Noise
     - ★ Analyze the hardness of a new problem by applying known attacks
  2. Convert into a public-key version
     - ★ $M + \sum \mathrm{Enc}(0)$ and leftover hash lemma over lattices

- Implementation result (128-bit security)

	Ctxt	PK	KeyGen	Enc	Dec	Add
Paillier	6144 bit	1.5 KB	437.39 s	62.46 ms	40.38 ms	12.40 μs
Ours	3072 bit	1.3 MB	0.35 s	0.72 ms	4.00 μs	0.40 μs

- Provide applications of our construction and general AHE

Ring Homomorphism: Chinese Remainder Theorem

Chinese Remainder Theorem

$$\begin{array}{rccc} \mathrm{CRT}_{(p_1,\ldots,p_k)}: & \prod_{i=1}^k \mathbb{Z}_{p_i} & \to & \mathbb{Z}_{\prod_{i=1}^k p_i} \\ & (m_1,\cdots,m_k) & \mapsto & m \end{array}$$

- $\mathbf{Enc}(m_1,\cdots,m_k) = \mathrm{CRT}_{(p_1,\ldots,p_k)}(m_1 + e_1 Q_1,\cdots,m_k + e_k Q_k)$
- At Eurocrypt 2013, Cheon et al. proposed a somewhat homomorphic encryption using this homomorphism.
- $\mathbf{Enc}(m_1,\cdots,m_k) = \mathrm{CRT}_{(q_0,p_1,\ldots,p_k)}(r, m_1 + e_1 Q_1,\cdots,m_k + e_k Q_k)$
- Semantically secure under the (extended)-ACD assumption

Ring Homomorphism: Chinese Remainder Theorem

Chinese Remainder Theorem

$$\begin{array}{rccc} \mathsf{CRT}_{(p_1,\ldots,p_k)}: & \prod_{i=1}^{k} \mathbb{Z}_{p_i} & \to & \mathbb{Z}_{\prod_{i=1}^{k} p_i} \\ & (m_1,\cdots,m_k) & \mapsto & m \end{array}$$

- **Enc**$(m_1,\cdots,m_k) = \mathsf{CRT}_{(p_1,\ldots,p_k)}(m_1+e_1Q_1,\cdots,m_k+e_kQ_k)$
- At Eurocrypt 2013, Cheon et al. proposed a somewhat homomorphic encryption using this homomorphism.
- **Enc**$(m_1,\cdots,m_k) = \mathsf{CRT}_{(q_0,p_1,\ldots,p_k)}(r,m_1+e_1Q_1,\cdots,m_k+e_kQ_k)$
- Semantically secure under the (extended)-ACD assumption

Ring Homomorphism: Chinese Remainder Theorem

Chinese Remainder Theorem

$$\begin{array}{rccc} \mathrm{CRT}_{(p_1,\ldots,p_k)}: & \prod_{i=1}^{k} \mathbb{Z}_{p_i} & \rightarrow & \mathbb{Z}_{\prod_{i=1}^{k} p_i} \\ & (m_1, \cdots, m_k) & \mapsto & m \end{array}$$

- $\mathbf{Enc}(m_1, \cdots, m_k) = \mathrm{CRT}_{(p_1,\ldots,p_k)}(m_1 + e_1 Q_1, \cdots, m_k + e_k Q_k)$
- At Eurocrypt 2013, Cheon et al. proposed a somewhat homomorphic encryption using this homomorphism.
- $\mathbf{Enc}(m_1, \cdots, m_k) = \mathrm{CRT}_{(q_0,p_1,\ldots,p_k)}(r, m_1 + e_1 Q_1, \cdots, m_k + e_k Q_k)$
- Semantically secure under the (extended)-ACD assumption

Ring Homomorphism: Chinese Remainder Theorem

Chinese Remainder Theorem

$$\begin{array}{rccc} \mathrm{CRT}_{(p_1,\ldots,p_k)}: & \prod_{i=1}^{k} \mathbb{Z}_{p_i} & \to & \mathbb{Z}_{\prod_{i=1}^{k} p_i} \\ & (m_1, \cdots, m_k) & \mapsto & m \end{array}$$

- $\textbf{Enc}(m_1, \cdots, m_k) = \mathrm{CRT}_{(p_1,\ldots,p_k)}(m_1 + e_1 Q_1, \cdots, m_k + e_k Q_k)$
- At Eurocrypt 2013, Cheon et al. proposed a somewhat homomorphic encryption using this homomorphism.
- $\textbf{Enc}(m_1, \cdots, m_k) = \mathrm{CRT}_{(q_0,p_1,\ldots,p_k)}(r, m_1 + e_1 Q_1, \cdots, m_k + e_k Q_k)$
- Semantically secure under the (extended)-ACD assumption

Ring Homomorphism: Chinese Remainder Theorem

Chinese Remainder Theorem

$$\begin{array}{rccc} \mathsf{CRT}_{(p_1,\ldots,p_k)}: & \prod_{i=1}^{k} \mathbb{Z}_{p_i} & \to & \mathbb{Z}_{\prod_{i=1}^{k} p_i} \\ & (m_1, \cdots, m_k) & \mapsto & m \end{array}$$

- $\textbf{Enc}(m_1, \cdots, m_k) = \mathsf{CRT}_{(p_1,\ldots,p_k)}(m_1 + e_1 Q_1, \cdots, m_k + e_k Q_k)$
- At Eurocrypt 2013, Cheon et al. proposed a somewhat homomorphic encryption using this homomorphism.
- $\textbf{Enc}(m_1, \cdots, m_k) = \mathsf{CRT}_{(q_0,p_1,\ldots,p_k)}(r, m_1 + e_1 Q_1, \cdots, m_k + e_k Q_k)$
- Semantically secure under the (extended)-ACD assumption

Ring Homomorphism: Inverse of a Homomorphism

- The inverse of a ring homomorphism is also a ring homomorphism!

Inverse of Chinese Remainder Theorem

$$\begin{array}{rccc} \mathrm{ModRed}_{(p_1,\ldots,p_k)}: & \mathbb{Z}_{\prod_{i=1}^k p_i} & \rightarrow & \prod_{i=1}^k \mathbb{Z}_{p_i} \\ & m & \mapsto & ([m]_{p_1}, \cdots, [m]_{p_k}) \end{array}$$

- With this homomorphism, we expect an efficient SHE where
  - the message space is comparable to Cheon et al.'s construction
  - the ciphertext size is smaller than Cheon et al.'s construction

Ring Homomorphism: Inverse of a Homomorphism

- The inverse of a ring homomorphism is also a ring homomorphism!

Inverse of Chinese Remainder Theorem

$$\begin{array}{rccc} \mathrm{ModRed}_{(p_1,\ldots,p_k)}: & \mathbb{Z}_{\prod_{i=1}^{k} p_i} & \rightarrow & \prod_{i=1}^{k} \mathbb{Z}_{p_i} \\ & m & \mapsto & ([m]_{p_1}, \cdots, [m]_{p_k}) \end{array}$$

- With this homomorphism, we expect an efficient SHE where
  - the message space is comparable to Cheon et al.'s construction
  - the ciphertext size is smaller than Cheon et al.'s construction

Ring Homomorphism: Inverse of a Homomorphism

- The inverse of a ring homomorphism is also a ring homomorphism!

Inverse of Chinese Remainder Theorem

$$\begin{array}{rccc} \mathrm{ModRed}_{(p_1,\ldots,p_k)}: & \mathbb{Z}_{\prod_{i=1}^k p_i} & \to & \prod_{i=1}^k \mathbb{Z}_{p_i} \\ & m & \mapsto & ([m]_{p_1}, \cdots, [m]_{p_k}) \end{array}$$

- With this homomorphism, we expect an efficient SHE where
  - the message space is comparable to Cheon et al.'s construction
  - the ciphertext size is smaller than Cheon et al.'s construction

Our Private-key Homomorphic Encryption Scheme (I)

- **Setup**(λ):
  - Choose η-bit distinct primes $p_1, \ldots, p_k$ satisfying $\gcd(Q, p_i) = 1$
  - $N := \prod_{i=1}^{k} p_i$
  - Output the private key $sk = (p_1, \ldots, p_k)$
- **Enc**(sk, m):
  - Randomly and uniformly choose e from $(-2^\rho, 2^\rho)$
  - Compute

$$\mathrm{ModRed}_{(p_1,\ldots,p_k)}(m + eQ) = \vec{c} = ([m + eQ]_{p_1}, \ldots, [m + eQ]_{p_k})$$

- **Dec**$(sk, \vec{c})$:
  - Compute

$$m = [\mathrm{CRT}_{(p_1,\ldots,p_k)}(\vec{c})]_Q = [m + eQ]_Q$$

Our Private-key Homomorphic Encryption Scheme (I)

- **Setup**(λ):
  - Choose η-bit distinct primes $p_1, \ldots, p_k$ satisfying $\gcd(Q, p_i) = 1$
  - $N := \prod_{i=1}^{k} p_i$
  - Output the private key $sk = (p_1, \ldots, p_k)$

- **Enc**(sk, m):
  - Randomly and uniformly choose e from $(-2^\rho, 2^\rho)$
  - Compute

$$\mathrm{ModRed}_{(p_1,\ldots,p_k)}(m + eQ) = \vec{c} = ([m + eQ]_{p_1}, \ldots, [m + eQ]_{p_k})$$

- **Dec**$(sk, \vec{c})$:
  - Compute

$$m = [\mathrm{CRT}_{(p_1,\ldots,p_k)}(\vec{c})]_Q = [m + eQ]_Q$$

Our Private-key Homomorphic Encryption Scheme (I)

- **Setup**(λ):
  - Choose η-bit distinct primes $p_1, \ldots, p_k$ satisfying $\gcd(Q, p_i) = 1$
  - $N := \prod_{i=1}^{k} p_i$
  - Output the private key $sk = (p_1, \ldots, p_k)$
- **Enc**(sk, m):
  - Randomly and uniformly choose e from $(-2^\rho, 2^\rho)$
  - Compute

$$\mathrm{ModRed}_{(p_1,\ldots,p_k)}(m + eQ) = \vec{c} = ([m + eQ]_{p_1}, \ldots, [m + eQ]_{p_k})$$

- **Dec**$(sk, \vec{c})$:
  - Compute

$$m = [\mathrm{CRT}_{(p_1,\ldots,p_k)}(\vec{c})]_Q = [m + eQ]_Q$$

Our Private-key Homomorphic Encryption Scheme (II)

- **Add**$(sk, \vec{c_1}, \vec{c_2})$: Output $\vec{c_1} + \vec{c_2}$ through the component-wise integer additions
- **Mul**$(sk, \vec{c_1}, \vec{c_2})$: Output $\vec{c_1} \times \vec{c_2}$ through the component-wise integer multiplications
- Correctness
  - $\vec{c}$: homomorphically generated ciphertext
  - $\vec{c} = \mathrm{ModRed}(f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q))$ for some f
  - If $|f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q)| < \frac{N}{2}$,

$$f(m_1+e_1 Q, \ldots, m_\ell+e_\ell Q) = \mathrm{CRT}(\mathrm{ModRed}(f(m_1+e_1 Q, \ldots, m_\ell+e_\ell Q)))$$

  - Otherwise,

$$\begin{aligned} f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q) &\neq ([f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q)]_N) \\ &= \mathrm{CRT}(\mathrm{ModRed}(f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q))) \end{aligned}$$

Our Private-key Homomorphic Encryption Scheme (II)

- **Add**$(sk, \vec{c_1}, \vec{c_2})$: Output $\vec{c_1} + \vec{c_2}$ through the component-wise integer additions
- **Mul**$(sk, \vec{c_1}, \vec{c_2})$: Output $\vec{c_1} \times \vec{c_2}$ through the component-wise integer multiplications
- Correctness
  - $\vec{c}$: homomorphically generated ciphertext
  - $\vec{c} = \mathrm{ModRed}(f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q))$ for some f
  - If $|f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q)| < \frac{N}{2}$,

$$f(m_1+e_1 Q, \ldots, m_\ell+e_\ell Q) = \mathrm{CRT}(\mathrm{ModRed}(f(m_1+e_1 Q, \ldots, m_\ell+e_\ell Q)))$$

  - Otherwise,

$$\begin{aligned} f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q) &\neq ([f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q)]_N) \\ &= \mathrm{CRT}(\mathrm{ModRed}(f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q))) \end{aligned}$$

Our Private-key Homomorphic Encryption Scheme (II)

- **Add**$(sk, \vec{c_1}, \vec{c_2})$: Output $\vec{c_1} + \vec{c_2}$ through the component-wise integer additions
- **Mul**$(sk, \vec{c_1}, \vec{c_2})$: Output $\vec{c_1} \times \vec{c_2}$ through the component-wise integer multiplications
- Correctness
  - $\vec{c}$: homomorphically generated ciphertext
  - $\vec{c} = \mathrm{ModRed}(f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q))$ for some f
  - If $|f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q)| < \frac{N}{2}$,

    $$f(m_1+e_1 Q, \ldots, m_\ell+e_\ell Q) = \mathrm{CRT}(\mathrm{ModRed}(f(m_1+e_1 Q, \ldots, m_\ell+e_\ell Q)))$$
  - Otherwise,

    $$\begin{aligned} f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q) &\neq ([f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q)]_N) \\ &= \mathrm{CRT}(\mathrm{ModRed}(f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q))) \end{aligned}$$

Our Private-key Homomorphic Encryption Scheme (II)

- **Add**$(sk, \vec{c_1}, \vec{c_2})$: Output $\vec{c_1} + \vec{c_2}$ through the component-wise integer additions
- **Mul**$(sk, \vec{c_1}, \vec{c_2})$: Output $\vec{c_1} \times \vec{c_2}$ through the component-wise integer multiplications
- Correctness
  - $\vec{c}$: homomorphically generated ciphertext
  - $\vec{c} = \mathrm{ModRed}(f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q))$ for some f
  - If $|f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q)| < \frac{N}{2}$,

    $$f(m_1+e_1 Q, \ldots, m_\ell+e_\ell Q) = \mathrm{CRT}(\mathrm{ModRed}(f(m_1+e_1 Q, \ldots, m_\ell+e_\ell Q)))$$

  - Otherwise,

    $$\begin{aligned} f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q) &\neq ([f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q)]_N) \\ &= \mathrm{CRT}(\mathrm{ModRed}(f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q))) \end{aligned}$$

Our Private-key Homomorphic Encryption Scheme (II)

- **Add**$(sk, \vec{c_1}, \vec{c_2})$: Output $\vec{c_1} + \vec{c_2}$ through the component-wise integer additions
- **Mul**$(sk, \vec{c_1}, \vec{c_2})$: Output $\vec{c_1} \times \vec{c_2}$ through the component-wise integer multiplications
- Correctness
  - $\vec{c}$: homomorphically generated ciphertext
  - $\vec{c} = \mathrm{ModRed}(f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q))$ for some f
  - If $|f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q)| < \frac{N}{2}$,

    $$f(m_1+e_1 Q, \ldots, m_\ell+e_\ell Q) = \mathrm{CRT}(\mathrm{ModRed}(f(m_1+e_1 Q, \ldots, m_\ell+e_\ell Q)))$$

  - Otherwise,

    $$\begin{aligned} f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q) &\neq ([f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q)]_N) \\ &= \mathrm{CRT}(\mathrm{ModRed}(f(m_1 + e_1 Q, \ldots, m_\ell + e_\ell Q))) \end{aligned}$$

The co-ACD Problem

Definition ($(\rho, \eta, 2; Q)$-co-ACD Problem)

- $\hat{\mathcal{D}}_{\rho,Q}(p_1, p_2) := \{\mathrm{ModRed}_{(p_1,p_2)}(eQ) | e \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho)\}$ for hidden η-bit primes p_j's.
- Given polynomially many samples from $\hat{\mathcal{D}}_{\rho,Q}(p_1, p_2)$, the $(\rho, \eta, 2; Q)$-co ACD problem is to find a certain p_j.

- The difference between the ACD problem and the co-ACD problem is the distribution that samples generated.

$$\mathcal{D}_\rho(p_1, p_2; Q_1, Q_2; q_0) := \{x = \mathrm{CRT}_{(q_0, p_1, \ldots, p_k)}(e_0, e_1 Q_1, e_2 Q_2) | \\ e_0 \leftarrow \mathbb{Z} \cap [0, q_0), e_i \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho)\}$$

The co-ACD Problem

Definition (($\rho, \eta, 2; Q$)-co-ACD Problem)

- $\hat{\mathcal{D}}_{\rho,Q}(p_1, p_2) := \{\mathsf{ModRed}_{(p_1,p_2)}(eQ) | e \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho)\}$ for hidden η-bit primes p_j's.
- Given polynomially many samples from $\hat{\mathcal{D}}_{\rho,Q}(p_1, p_2)$, the $(\rho, \eta, 2; Q)$-co ACD problem is to find a certain p_j.

- The difference between the ACD problem and the co-ACD problem is the distribution that samples generated.

$$\mathcal{D}_\rho(p_1, p_2; Q_1, Q_2; q_0) := \{x = \mathsf{CRT}_{(q_0, p_1, \ldots, p_k)}(e_0, e_1 Q_1, e_2 Q_2) | \\ e_0 \leftarrow \mathbb{Z} \cap [0, q_0), e_i \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho)\}$$

The co-ACD Problem

Definition ($(\rho, \eta, 2; Q)$-co-ACD Problem)

- $\hat{\mathcal{D}}_{\rho,Q}(p_1, p_2) := \{\mathrm{ModRed}_{(p_1,p_2)}(eQ) | e \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho)\}$ for hidden η-bit primes p_j's.
- Given polynomially many samples from $\hat{\mathcal{D}}_{\rho,Q}(p_1, p_2)$, the $(\rho, \eta, 2; Q)$-co ACD problem is to find a certain p_j.

- The difference between the ACD problem and the co-ACD problem is the distribution that samples generated.

$$\begin{aligned}\mathcal{D}_\rho(p_1, p_2; Q_1, Q_2; q_0) := \{x = \mathrm{CRT}_{(q_0,p_1,\ldots,p_k)}(e_0, e_1 Q_1, e_2 Q_2)| \\ e_0 \leftarrow \mathbb{Z} \cap [0, q_0), e_i \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho)\}\end{aligned}$$

Security of Our Construction

Decisional version

Given polynomially many samples from $\hat{\mathcal{D}}_{\rho,Q}$ and the uniform distribution on $\mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2}$, determine whether the target vector $\vec{x}$ is sampled from $\hat{\mathcal{D}}_{\rho,Q}$ or the uniform distribution on $\mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2}$.

- Our scheme is semantically secure under the assumption that the decisional version of the $(\rho, \eta, k; Q)$-co-ACD problem is hard.
- There is no reduction between the co-ACD assumption and other well-known cryptographic assumptions.
- To show the hardness of the co-ACD problem, apply known attacks to solve the co-ACD problem.

Security of Our Construction

Decisional version

Given polynomially many samples from $\hat{\mathcal{D}}_{\rho,Q}$ and the uniform distribution on $\mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2}$, determine whether the target vector $\vec{x}$ is sampled from $\hat{\mathcal{D}}_{\rho,Q}$ or the uniform distribution on $\mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2}$.

- Our scheme is semantically secure under the assumption that the decisional version of the $(\rho, \eta, k; Q)$-co-ACD problem is hard.
- There is no reduction between the co-ACD assumption and other well-known cryptographic assumptions.
- To show the hardness of the co-ACD problem, apply known attacks to solve the co-ACD problem.

Security of Our Construction

Decisional version

Given polynomially many samples from $\hat{\mathcal{D}}_{\rho,Q}$ and the uniform distribution on $\mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2}$, determine whether the target vector $\vec{x}$ is sampled from $\hat{\mathcal{D}}_{\rho,Q}$ or the uniform distribution on $\mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2}$.

- Our scheme is semantically secure under the assumption that the decisional version of the $(\rho, \eta, k; Q)$-co-ACD problem is hard.
- There is no reduction between the co-ACD assumption and other well-known cryptographic assumptions.
- To show the hardness of the co-ACD problem, apply known attacks to solve the co-ACD problem.

Analysis of the Hardness of the co-ACD Problem

Simplified co-ACD Problem

Given many samples $\mathrm{ModRed}_{(p_1,\ldots,p_k)}(e_i Q) := (e_i Q \bmod p_j)_{1\le j\le k}$ for η-bit hidden primes p_j's and an integer $e_i \in (-2^\rho, 2^\rho)$, the co-ACD problem is to find some prime p_j.

- Using one component
  - Statistical distance from the uniform distribution: $\rho > \eta + \lambda$
  - Chen-Ngyuen's attack: $\rho > 2\lambda$
- Using multiple components
  - Coppersmith attack: $\rho > \eta + \lambda$
  - Orthogonal lattice attack: $\rho > (k-1)\eta$

Analysis of the Hardness of the co-ACD Problem

Simplified co-ACD Problem

Given many samples $\mathrm{ModRed}_{(p_1,\ldots,p_k)}(e_i Q) := (e_i Q \bmod p_j)_{1\le j\le k}$ for η-bit hidden primes p_j's and an integer $e_i \in (-2^\rho, 2^\rho)$, the co-ACD problem is to find some prime p_j.

- Using one component
  - Statistical distance from the uniform distribution: $\rho > \eta + \lambda$
  - Chen-Ngyuen's attack: $\rho > 2\lambda$
- Using multiple components
  - Coppersmith attack: $\rho > \eta + \lambda$
  - Orthogonal lattice attack: $\rho > (k-1)\eta$

Analysis of the Hardness of the co-ACD Problem

Simplified co-ACD Problem

Given many samples $\mathsf{ModRed}_{(p_1,\ldots,p_k)}(e_i Q) := (e_i Q \bmod p_j)_{1\le j\le k}$ for η-bit hidden primes p_j's and an integer $e_i \in (-2^\rho, 2^\rho)$, the co-ACD problem is to find some prime p_j.

- Using one component
  - Statistical distance from the uniform distribution: $\rho > \eta + \lambda$
  - Chen-Ngyuen's attack: $\rho > 2\lambda$
- Using multiple components
  - Coppersmith attack: $\rho > \eta + \lambda$
  - Orthogonal lattice attack: $\rho > (k-1)\eta$

Parameters of Our Private-key Scheme

- Parameters
  - $\eta = O(\lambda^2)$ to resist factoring attack of N
  - $\rho > \eta + \lambda$ to avoid Coppersmith's attack
  - $\rho > (k-1)\eta$ to avoid orthogonal lattice attack
  - $k = 2$ for the efficiency
- The bit size ρ of noise is too large to support a multiplication.
- For correct decryption, $(m_1 + e_1 Q) \times (m_2 + e_2 Q)$ is less than $\frac{N}{2}$.
- However,

$$(m_1 + e_1 Q) \times (m_2 + e_2 Q) \approx 2^{2\rho} > 2^{2(k-1)\eta} > 2^{k\eta} \approx N$$

- As a result, we obtain an efficient private-key AHE where
  - the ciphertext size is smaller than Paillier
  - the computational cost is lower than Paillier

Parameters of Our Private-key Scheme

- Parameters
  - $\eta = O(\lambda^2)$ to resist factoring attack of N
  - $\rho > \eta + \lambda$ to avoid Coppersmith's attack
  - $\rho > (k-1)\eta$ to avoid orthogonal lattice attack
  - $k = 2$ for the efficiency
- The bit size ρ of noise is too large to support a multiplication.
- For correct decryption, $(m_1 + e_1 Q) \times (m_2 + e_2 Q)$ is less than $\frac{N}{2}$.
- However,

$$(m_1 + e_1 Q) \times (m_2 + e_2 Q) \approx 2^{2\rho} > 2^{2(k-1)\eta} > 2^{k\eta} \approx N$$

- As a result, we obtain an efficient private-key AHE where
  - the ciphertext size is smaller than Paillier
  - the computational cost is lower than Paillier

Parameters of Our Private-key Scheme

- Parameters
  - $\eta = O(\lambda^2)$ to resist factoring attack of N
  - $\rho > \eta + \lambda$ to avoid Coppersmith's attack
  - $\rho > (k-1)\eta$ to avoid orthogonal lattice attack
  - $k = 2$ for the efficiency
- The bit size ρ of noise is too large to support a multiplication.
- For correct decryption, $(m_1 + e_1 Q) \times (m_2 + e_2 Q)$ is less than $\frac{N}{2}$.
- However,

$$(m_1 + e_1 Q) \times (m_2 + e_2 Q) \approx 2^{2\rho} > 2^{2(k-1)\eta} > 2^{k\eta} \approx N$$

- As a result, we obtain an efficient private-key AHE where
  - the ciphertext size is smaller than Paillier
  - the computational cost is lower than Paillier

Parameters of Our Private-key Scheme

- Parameters
  - $\eta = O(\lambda^2)$ to resist factoring attack of N
  - $\rho > \eta + \lambda$ to avoid Coppersmith's attack
  - $\rho > (k-1)\eta$ to avoid orthogonal lattice attack
  - $k = 2$ for the efficiency
- The bit size ρ of noise is too large to support a multiplication.
- For correct decryption, $(m_1 + e_1 Q) \times (m_2 + e_2 Q)$ is less than $\frac{N}{2}$.
- However,

$$(m_1 + e_1 Q) \times (m_2 + e_2 Q) \approx 2^{2\rho} > 2^{2(k-1)\eta} > 2^{k\eta} \approx N$$

- As a result, we obtain an efficient private-key AHE where
  - the ciphertext size is smaller than Paillier
  - the computational cost is lower than Paillier

Parameters of Our Private-key Scheme

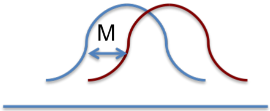
- Parameters
  - $\eta = O(\lambda^2)$ to resist factoring attack of N
  - $\rho > \eta + \lambda$ to avoid Coppersmith's attack
  - $\rho > (k-1)\eta$ to avoid orthogonal lattice attack
  - $k = 2$ for the efficiency
- The bit size ρ of noise is too large to support a multiplication.
- For correct decryption, $(m_1 + e_1 Q) \times (m_2 + e_2 Q)$ is less than $\frac{N}{2}$.
- However,

$$(m_1 + e_1 Q) \times (m_2 + e_2 Q) \approx 2^{2\rho} > 2^{2(k-1)\eta} > 2^{k\eta} \approx N$$

- As a result, we obtain an efficient private-key AHE where
  - the ciphertext size is smaller than Paillier
  - the computational cost is lower than Paillier

Convert into a Public-key Version

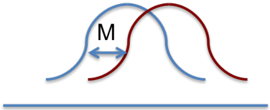
- The distribution of $\sum \mathrm{Enc}(0)$ and $\mathrm{M}+\sum \mathrm{Enc}(0)$



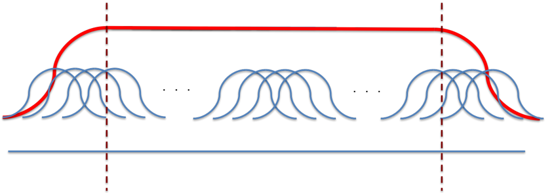
- By shifting the subset-sum of Enc(0)'s, we can obtain the following distribution: $\sum_{j=1}^{m} s_j \mathrm{Enc}_j(0) + \sum_{i=1}^{n} t_i \mathrm{Enc}_i(0)$ where $s_j \leftarrow \{0,1\}$ and $t_i \leftarrow [0, 2^{\mu})$

Convert into a Public-key Version

- The distribution of $\sum \mathsf{Enc}(0)$ and $\mathsf{M}+\sum \mathsf{Enc}(0)$

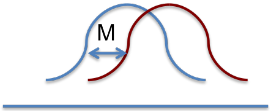


- By shifting the subset-sum of Enc(0)'s, we can obtain the following distribution: $\sum_{j=1}^{m} s_j \mathsf{Enc}_j(0) + \sum_{i=1}^{n} t_i \mathsf{Enc}_i(0)$ where $s_j \leftarrow \{0,1\}$ and $t_i \leftarrow [0, 2^{\mu})$

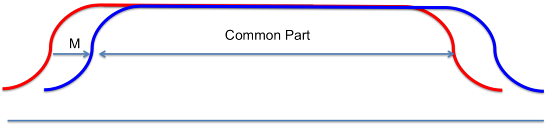


Convert into a Public-key Version

- The distribution of $\sum \mathsf{Enc}(0)$ and $\mathsf{M}+\sum \mathsf{Enc}(0)$



- By shifting the subset-sum of Enc(0)'s, we can obtain the following distribution: $\sum_{j=1}^{m} s_j \mathsf{Enc}_j(0) + \sum_{i=1}^{n} t_i \mathsf{Enc}_i(0)$ where $s_j \leftarrow \{0,1\}$ and $t_i \leftarrow [0, 2^{\mu})$



Leftover Hash Lemma over Lattices

Lemma (Leftover Hash Lemma over Lattices; CLT13)

- $L \subset \mathbb{Z}^n$: *a lattice of rank* n *of a basis* $\mathbf{B} = (\vec{b}_1, \ldots, \vec{b}_n)$
- $\mathcal{D}_{\mathbf{B}}$: *a distribution of outputting a random element sampled from the half-open parallelepiped generated by* $\mathbf{B}$
- $x_i \leftarrow \mathcal{D}_{\mathbf{B}}$ *for* $1 \leq i \leq m$
- $\vec{y} = \sum_{j=1}^{m} s_j \vec{x}_j + \sum_{i=1}^{n} t_i \vec{b}_i$ *where* $s_j \leftarrow \{0, 1\}$ *and* $t_i \leftarrow [0, 2^{\mu}) \cap \mathbb{Z}$
- $\vec{y}' \leftarrow \mathcal{D}_{2^{\mu}\mathbf{B}}$ *for* $2^{\mu}\mathbf{B} = (2^{\mu}\vec{b}_1, \ldots, 2^{\mu}\vec{b}_n)$

$\Longrightarrow (\vec{x}_1, \ldots, \vec{x}_m, \vec{y})$ *and* $(\vec{x}_1, \ldots, \vec{x}_m, \vec{y}')$ *are* ϵ*-statistically close, with* $\epsilon = \frac{mn}{2^{\mu}} + \frac{1}{2} \cdot \sqrt{\frac{|\det L|}{2^m}}$.

e.g.) $\eta = 1536$ $(\Rightarrow |\det L| \leq 2^{3072})$, $m = 3328$, $n = 2$, $\mu = 142$, $\epsilon < 2^{-128}$

Public-key Version of Our Scheme

- **Setup**(1^λ):
  - $\vec{b}_1 = \mathrm{ModRed}_{(p_1,p_2)}(e_1'Q)$ and $\vec{b}_2 = \mathrm{ModRed}_{(p_1,p_2)}(e_2'Q)$ so that the determinant of the lattice generated by $\vec{b}_1$ and $\vec{b}_2$ are sufficiently large.
  - $\vec{x}_j = \mathrm{ModRed}_{(p_1,p_2)}(e_jQ)$ for $1 \le j \le m$ which are contained in the half-open parallelepiped generated by $\vec{b}_1$ and $\vec{b}_2$.
  - $pk = \{Q, \vec{b}_1, \vec{b}_2, \vec{x}_1, \ldots, \vec{x}_m\}$ and $sk = \{p_1, p_2\}$, where $\mathbb{Z}_Q$ is the message space.

- **Enc**(pk, M):
  - Choose $s_j \leftarrow \{0,1\}$, $t_i \leftarrow [0, 2^\mu) \cap \mathbb{Z}$ for $j \in \{1, \ldots, m\}$ and $i \in \{0,1\}$.
  - Compute

$$\vec{c} = (M, M) + \sum_{j=1}^{m} s_j \vec{x}_j + \sum_{i=1}^{2} t_i \vec{b}_i,$$

where '$+$' is a binary operation meaning an addition in $\mathbb{Z} \times \mathbb{Z}$.

Public-key Version of Our Scheme

- **Setup**(1^λ):
  - $\vec{b}_1 = \mathrm{ModRed}_{(p_1,p_2)}(e'_1 Q)$ and $\vec{b}_2 = \mathrm{ModRed}_{(p_1,p_2)}(e'_2 Q)$ so that the determinant of the lattice generated by $\vec{b}_1$ and $\vec{b}_2$ are sufficiently large.
  - $\vec{x}_j = \mathrm{ModRed}_{(p_1,p_2)}(e_j Q)$ for $1 \le j \le m$ which are contained in the half-open parallelepiped generated by $\vec{b}_1$ and $\vec{b}_2$.
  - $pk = \{Q, \vec{b}_1, \vec{b}_2, \vec{x}_1, \ldots, \vec{x}_m\}$ and $sk = \{p_1, p_2\}$, where $\mathbb{Z}_Q$ is the message space.

- **Enc**(pk, M):
  - Choose $s_j \leftarrow \{0,1\}$, $t_i \leftarrow [0, 2^\mu) \cap \mathbb{Z}$ for $j \in \{1, \ldots, m\}$ and $i \in \{0,1\}$.
  - Compute

$$\vec{c} = (M, M) + \sum_{j=1}^{m} s_j \vec{x}_j + \sum_{i=1}^{2} t_i \vec{b}_i,$$

where '$+$' is a binary operation meaning an addition in $\mathbb{Z} \times \mathbb{Z}$.

Efficiency: Parameter Sizes

Table: Parameter Sizes

	λ	η	ρ	m	μ	$\log Q$	$\log A$	γ	PK
Pai99	128	1536	–	–	–	3072	∞	6144	1.5 KB
NLV11	120	–	–	–	–	10	20	61440	7.6 KB
JL13	128	1536	–	–	–	256	∞	3072	0.8 KB
Ours	128	1536	1792	3328	142	256	1134	3072	1.3 MB
		2194	2450	4645			1536	4388	2.6 MB
		2706	2962	5659			2048	5412	3.9 MB

Efficiency: Implementation Results

- System: Intel Core i7-2600 CPU running at 3.4 GHz with 16 GB RAM

Table: Parameter Sizes, Implementation Results, and Comparison

	λ	$\log A$	Setup	Enc	Dec	Add
Pai99	128	∞	437.39 s	62.46 ms	40.38 ms	12.40 μs
NLV11†	120	20	0.11 s	164.00 ms	4.00 ms	$\leq$ 1.00 ms
JL13	128	∞	250.32 s	2.07 ms	903.36 ms	2.40 μs
Ours	128	1134	0.35 s	0.72 ms	4.00 μs	0.40 μs
		1536	1.18 s	1.07 ms	8.00 μs	0.80 μs
		2048	2.34 s	1.29 ms	8.80 μs	0.80 μs

† We referred to the implementation results in [NLV11] and they were done on a 2.1 GHz Intel Core 2 Duo, with 3 MB L3 cache and 1 GB of memory.

Applications

- Symmetric polynomial evaluation
  - A symmetric polynomial of degree ($d < n$) in n variables can be represented by the sum of power-sum polynomials of degree at most d.
  - Modify an encryption algorithm by

$$\mathcal{E}_d(pk, M) := (\mathbf{Enc}(pk, M), \mathbf{Enc}(pk, M^2), \ldots, \mathbf{Enc}(pk, M^d))$$

  - Compute the variance of 1000 128-bit integers: 120 μs

- Private set operations based on a polynomial representation of a set
  - Polynomial representation of $S = \{s_1, \cdots, s_\ell\}$: $f_S(x) = \prod_{i=1}^{\ell}(x - s_i)$
  - To recover a set from a polynomial: Need a root finding algorithm
    - ★ For a root finding algorithm, the message space should be a field.
    - ★ Previous additive homomorphic encryption scheme: $\mathbb{Z}_\sigma$ for a composite or hidden prime σ
    - ★ The message space of our scheme can be a field.

Applications

- Symmetric polynomial evaluation
  - A symmetric polynomial of degree ($d < n$) in n variables can be represented by the sum of power-sum polynomials of degree at most d.
  - Modify an encryption algorithm by

$$\mathcal{E}_d(pk, M) := (\mathbf{Enc}(pk, M), \mathbf{Enc}(pk, M^2), \ldots, \mathbf{Enc}(pk, M^d))$$

  - Compute the variance of 1000 128-bit integers: 120 μs

- Private set operations based on a polynomial representation of a set
  - Polynomial representation of $S = \{s_1, \cdots, s_\ell\}$: $f_S(x) = \prod_{i=1}^{\ell}(x - s_i)$
  - To recover a set from a polynomial: Need a root finding algorithm
    - ★ For a root finding algorithm, the message space should be a field.
    - ★ Previous additive homomorphic encryption scheme: $\mathbb{Z}_\sigma$ for a composite or hidden prime σ
    - ★ The message space of our scheme can be a field.

Conclusions & Further Works

- Provide an efficient AHE scheme based on the new assumption
- Study on the co-ACD problem
  - More analysis of the hardness of the co-ACD problem
  - Relation between the computational version and decisional version
- IND-CCA2 PKE using Fujisaki-Okamoto conversion
  - Compared to RSA-OAEP, enc: 6 times slower, dec: 1000 times faster[1]
- Reduce the ciphertext size excluding the factoring assumption
  - Ciphertext: 3072 bits $\Rightarrow$ 800+2 $\times$ ($\log Q + \log A$) bits
- Generalize leftover hash lemma using large coefficients
  - PK Size: 1.3 MB $\Rightarrow$ 3.3 KB ($\nu = 1000$ where $s_j \leftarrow [0, 2^\nu)$)

[1] Crypto++ Library 5.6.2, available at http://www.cryptopp.com

Conclusions & Further Works

- Provide an efficient AHE scheme based on the new assumption
- Study on the co-ACD problem
  - More analysis of the hardness of the co-ACD problem
  - Relation between the computational version and decisional version
- IND-CCA2 PKE using Fujisaki-Okamoto conversion
  - Compared to RSA-OAEP, enc: 6 times slower, dec: 1000 times faster[1]
- Reduce the ciphertext size excluding the factoring assumption
  - Ciphertext: 3072 bits $\Rightarrow$ 800+2 $\times$ ($\log Q + \log A$) bits
- Generalize leftover hash lemma using large coefficients
  - PK Size: 1.3 MB $\Rightarrow$ 3.3 KB ($\nu = 1000$ where $s_j \leftarrow [0, 2^{\nu})$)

[1] Crypto++ Library 5.6.2, available at http://www.cryptopp.com

Conclusions & Further Works

- Provide an efficient AHE scheme based on the new assumption
- Study on the co-ACD problem
  - More analysis of the hardness of the co-ACD problem
  - Relation between the computational version and decisional version
- IND-CCA2 PKE using Fujisaki-Okamoto conversion
  - Compared to RSA-OAEP, enc: 6 times slower, dec: 1000 times faster[1]
- Reduce the ciphertext size excluding the factoring assumption
  - Ciphertext: 3072 bits $\Rightarrow$ 800+2 $\times$ ($\log Q + \log A$) bits
- Generalize leftover hash lemma using large coefficients
  - PK Size: 1.3 MB $\Rightarrow$ 3.3 KB ($\nu = 1000$ where $s_j \leftarrow [0, 2^\nu)$)

[1] Crypto++ Library 5.6.2, available at http://www.cryptopp.com

Conclusions & Further Works

- Provide an efficient AHE scheme based on the new assumption
- Study on the co-ACD problem
  - More analysis of the hardness of the co-ACD problem
  - Relation between the computational version and decisional version
- IND-CCA2 PKE using Fujisaki-Okamoto conversion
  - Compared to RSA-OAEP, enc: 6 times slower, dec: 1000 times faster[1]
- Reduce the ciphertext size excluding the factoring assumption
  - Ciphertext: 3072 bits $\Rightarrow$ 800+2 $\times$ ($\log Q + \log A$) bits
- Generalize leftover hash lemma using large coefficients
  - PK Size: 1.3 MB $\Rightarrow$ 3.3 KB ($\nu = 1000$ where $s_j \leftarrow [0, 2^\nu)$)

[1] Crypto++ Library 5.6.2, available at http://www.cryptopp.com

Conclusions & Further Works

- Provide an efficient AHE scheme based on the new assumption
- Study on the co-ACD problem
  - More analysis of the hardness of the co-ACD problem
  - Relation between the computational version and decisional version
- IND-CCA2 PKE using Fujisaki-Okamoto conversion
  - Compared to RSA-OAEP, enc: 6 times slower, dec: 1000 times faster[1]
- Reduce the ciphertext size excluding the factoring assumption
  - Ciphertext: 3072 bits $\Rightarrow$ 800+2 $\times$ ($\log Q + \log A$) bits
- Generalize leftover hash lemma using large coefficients
  - PK Size: 1.3 MB $\Rightarrow$ 3.3 KB ($\nu = 1000$ where $s_j \leftarrow [0, 2^{\nu})$)

[1] Crypto++ Library 5.6.2, available at http://www.cryptopp.com

Conclusions & Further Works

- Provide an efficient AHE scheme based on the new assumption
- Study on the co-ACD problem
  - More analysis of the hardness of the co-ACD problem
  - Relation between the computational version and decisional version
- IND-CCA2 PKE using Fujisaki-Okamoto conversion
  - Compared to RSA-OAEP, enc: 6 times slower, dec: 1000 times faster[1]
- Reduce the ciphertext size excluding the factoring assumption
  - Ciphertext: 3072 bits $\Rightarrow$ 800+2 $\times$ ($\log Q + \log A$) bits
- Generalize leftover hash lemma using large coefficients
  - PK Size: 1.3 MB $\Rightarrow$ 3.3 KB ($\nu = 1000$ where $s_j \leftarrow [0, 2^\nu)$)

***** Thanks and Any Question?*****

[1] Crypto++ Library 5.6.2, available at http://www.cryptopp.com