A Study on Partially Homomorphic Encryption Schemes

Shifat P. Mithila
Advisor: Dr. Koray Karabina

Florida Atlantic University

smithila2014@fau.com

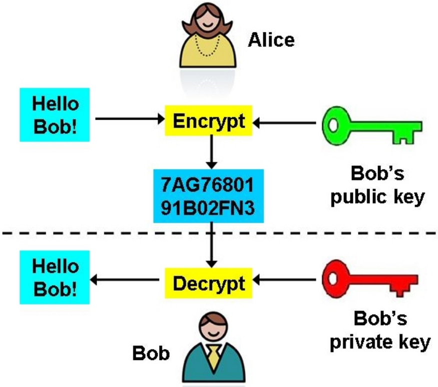
November 15, 2017

Overview

1. Public key encryption schemes
2. Homomorphic encryption schemes
3. CGS encryption scheme
4. Boosting technique for linearly homomorphic encryption scheme
5. Concluding remarks

Public key encryption scheme

- Symmetric key cryptography:
  $\rightarrow$ one single key used.
- Asymmetric/ Public key cryptography:
  $\rightarrow$ a pair of keys (pk, sk) is used.



Short history of public key encryption

$\rightarrow$ Introduced in 1976, by Diffie and Hellman.
$\rightarrow$ Diffie and Hellman proposed "key-exchange protocol".

RSA scheme: Ron Rivest, Adi Shamir and Leonard Adleman, in 1978

- First public key cryptosystem.
- Based on integer factorization problem.
- Security depends on: Factoring N, computing $\phi(N)$ or computing d.
- Widely used in secure data transmission, mostly in "key agreement" and "digital signature".

ElGamal encryption scheme: Construction

ElGamal scheme: Taher ElGamal, in 1985

- Based on Diffie-Hellman key exchange.
- Implemented on hybrid cryptosystems, PGP, free GNU privacy guard software etc.

KeyGen :

- Input is $(\mathbb{G}, q, g)$.
- Choose a random $a \longleftarrow [1, q-1]$
- Compute g^a
- Outputs are the public key is $\langle \mathbb{G}, q, g, g^a \rangle$ and the private key is $\langle \mathbb{G}, q, g, a \rangle$

ElGamal encryption scheme: Construction

Enc :

- Input a public key $pk = \langle \mathbb{G}, q, g, g^a \rangle$ and a message $m \in \mathbb{G}$
- Choose a random $r \longleftarrow [1, q]$
- Output the ciphertext $(c_1, c_2) := (g^r, (g^a)^r \cdot m)$

Dec :

- Input a private key $sk = \langle \mathbb{G}, q, g, a \rangle$ and a ciphertext (c_1, c_2)
- Output the message $m := c_2 / {c_1}^a$

Security of ElGamal scheme

- Breaking ElGamal $\equiv$ Computational Diffie-Hellman (CDH) problem. (**CDH:** From given (g, g^a, g^b), can we find g^{ab} having no knowledge about a and b?).

- Semantic security of the ElGamal $\equiv$ Decisional Diffie-Hellman (DDH) problem.

  (**DDH:** Can we distinguish between the given tuples (g, g^a, g^b, g^{ab}) and (g, g^a, g^b, g^r), having no knowledge about a and b?).

Effieciency of ElGamal scheme

Table: **Efficiency of ElGamal Encryption Scheme**

Functions	Operations (we denote multiplication by M, pseudo random number generation by PRNG, scalar multiplication by SM, division by D, subtraction by S, addition by A and exponentiation by E)
ElGamal Key Generation	1PRNG + 1E
ElGamal Encrytion	1PRNG + 2E + 1M
ElGamal Decrytion	1E + 1D

Homomorphic encryption scheme

- Certain computations (addition and multiplication) can be performed on the encrypted plaintexts/ ciphertexts.
- Generates a ciphertext such that when decrypted, gives same result from the similar operations performed on the plaintexts.
- Outsourcing computations.
- Implemented in cloud computing, electronic voting protocol, watermarking and fingerprinting, secure multiparty computations etc.
- Message space $\mathcal{M} = (\mathbb{Z}, +, \cdot)$

Properties of homomorphic encryption scheme

Additively homomorphic:

$$\text{Enc}(m_1 + m_2) = \text{Enc}(m_1) \boxplus \text{Enc}(m_2)$$

Multiplicatively homomorphic:

$$\text{Enc}(m_1 \cdot m_2) = \text{Enc}(m_1) \boxdot \text{Enc}(m_2)$$

Scaler multiplication property:

$$\begin{aligned}
\text{Enc}(s \cdot m) &= \text{Enc}(m + m + \cdots + m) \\
&= \text{Enc}(m) \boxplus \text{Enc}(m) \boxplus \cdots \boxplus \text{Enc}(m) \\
&= s \boxdot \text{Enc}(m)
\end{aligned}$$

Different types of homomorphic scheme

Partially homomorphic scheme:

- Allows only one homomorphic property (addition or multiplication but not both).

Fully homomorphic scheme:

- Allows both the homomorphic properties (arbitrary number of additions and multiplications).

Somewhat homomorphic scheme:

- More than partilly homomorphic.
- But Not fully homomorphic.

Examples of homomorphic schemes

- **RSA is partially (multiplicative) homomorphic**
- **ElGamal is partially (multiplicative) homomorphic**

$$\begin{aligned} E(m_1) \boxdot E(m_2) &= (g^{r_1}, (g^a)^{r_1} \cdot m_1) \boxdot (g^{r_2}, (g^a)^{r_2} \cdot m_2) \\ &= (g^{r_1+r_2}, (g^a)^{r_1+r_2} \cdot m_1 \cdot m_2) \\ &= (g^r, (g^a)^r \cdot m_1 \cdot m_2) \\ &= E(m_1 \cdot m_2) \end{aligned}$$

for some $r = r_1 + r_2$

CGS homomorphic encryption scheme

- Cramer, Genarro and Schoenmakers, in 1997.
- Presented as a variant on the ElGamal scheme.
- Consists of four faces: key generation, encryption, evaluation functions and decryption.
  $\mathsf{CGS} = (\mathsf{KeyGen}_{\mathsf{CGS}}, \mathsf{Enc}_{\mathsf{CGS}}, \mathsf{Dec}_{\mathsf{CGS}}, \mathsf{Eval}_{\mathsf{CGS}})$
- Linearly homomorphic scheme.

CGS encryption scheme: Construction

$\text{KeyGen}_{\text{CGS}}$:

- Inputs are security parameter 1^n, group $\mathbb{G}$ and element $g \in \mathbb{G}$.
- Choose a random $a \longleftarrow [1, q-1]$
- Compute g^a
- Outputs are the private key $sk = a$, public key $pk = g^a$

Enc_{CGS} :

- Inputs are public key $G = g^a$, message $m \in [-B, B]$
- Choose a random number $r \in_R [1, q-1]$
- If r is prime then
  compute $x := g^r$
  compute $y := G^r * G^m$
- Output the ciphertext $c = [x, y]$

CGS encryption scheme: Construction

Dec$_{CGS}$:

- Inputs are secret key a, the ciphertext $c = [x, y]$
- Compute $k_1 := x^a$
- Compute $k_2 := y/k_1$
- For $i \in [-B, B]$
  If $G^i == k_2$ then $m = i$
  Otherwise return "error"
- Output the message m

CGS encryption scheme: Homomorphic properties (Eval_{CGS})

Addition (Add_{CGS})

- Inputs are the ciphertext pair $c_1 = \text{Enc}_{\text{CGS}}(m_1) = [x_1, y_1]$ and $c_2 = \text{Enc}_{\text{CGS}}(m_2) = [x_2, y_2]$
- Compute $x := x_1 \cdot x_2$
- Compute $y := y_1 \cdot y_2$
- Output the ciphertext $c = c_1 \boxplus c_2 = [x, y] = \text{Enc}_{\text{CGS}}(m_1 + m_2)$

Scaler multiplication ($SMult_{CGS}$)

- Inputs are the ciphertext $c_1 = [x_1, y_1]$ and a scaler $s \in \mathcal{M}$
- Compute $x := x_1^s$
- Compute $y := y_1^s$
- Output is the ciphertext $s \boxdot c_1 = [x, y] = \text{Enc}_{\text{CGS}}(s \cdot m_1)$

CGS encryption scheme: Homomorphic properties (Eval_{CGS})

Linear Combination ($LinComb_{CGS}$)

- Inputs are a pair of sets (s, c) where $s = [s_1, s_2 .....]$ and $c = [c_1, c_2, ...]$ where each $c_i = [x_i, y_i]$
- Define $k := \#s$
- Choose $x := 1$
- Choose $y := 1$
- For i=1 to k
  Compute $x := x.x_i^{s_i}$
  Compute $y := y.y_i^{s_i}$
- Output is the ciphertext $c = [x, y] = \text{Enc}_{\text{CGS}}(\sum_i s_i \cdot m_i)$

Security of CGS scheme

- Discrete logarithm problem is required to be intractable.
- Computational Diffie-Hellman problem has to be intractable.
- Semantic security of the CGS encryption scheme requires the intractability of the decisional Diffie-Hellman(DDH) problem.
- Not known whether the security of ElGamal and CGS schemes are equivalent or not.

Efficiency of CGS scheme

Table: **Efficiency of CGS Encryption Scheme**

Functions	Operations
CGS Key generation	1 PRNG + 1E
CGS Encrytion	1 PRPNG + 3E + 1M
CGS Decrytion	2E + 1D
CGS Addition	2M
CGS Scalar multiplication	2E
CGS Linear combination ($k = \#s$)	2k E + 2k M

Boosting linearly homomorphic encryption scheme

- Dario Catalano and Dario Fiore, 2015.
- Converts a public-space LHE scheme $\widehat{\mathsf{HE}} = (\widehat{\mathsf{KeyGen}}, \widehat{\mathsf{Enc}}, \widehat{\mathsf{Eval}}, \widehat{\mathsf{Dec}})$ to a HE scheme supporting one multiplication, denoted by $\mathsf{HE_B} = (\mathsf{KeyGen_B}, \mathsf{Enc_B}, \mathsf{Eval_B}, \mathsf{Dec_B})$.
- The message space $\rightarrow$ public ring.
- Claimed to work on virtually all the existing number theoretic LHE such as Paillier, ElGamal or Goldwasser-Micalli.

Boosting LHE scheme: Construction

$\mathsf{KeyGen_B}$:

$\widehat{\mathsf{KeyGen}} = \mathsf{KeyGen_B}$.

$\mathsf{Enc_B}$:

- Inputs are public key $pk = G$, message m
- Choose a random number $b \in_R \mathcal{M}$
- Compute $u = m - b$
- Compute $\beta = \widehat{\mathsf{Enc}}(b)$
- Output the ciphertext $c = [u, \beta]$

Boosting LHE scheme: Construction

Evaluation functions for the Boosted-LHE scheme ($\mathsf{Eval_B}$)

- Ciphertexts are of two levels:
  $\rightarrow$ **Level 1 ciphertext** : encode "fresh" messages/ linear combinations of "fresh" messages.

  $\rightarrow$ **Level 2 ciphertexts** : "multiplied" level 1 ciphertexts.

- Five different evaluation functions:

  $\rightarrow$ Add_1: Addition between two level 1 ciphertexts.

  $\rightarrow$ Mult_1: Multiplication between two level 1 ciphertexts.

  $\rightarrow$ Add_2: Addition between two level 2 ciphertexts.

  $\rightarrow$ SMult_1: Scalar multiplication over a single level 1 ciphertext.

  $\rightarrow$ SMult_2: Scalar multiplication over a single level 2 ciphertext.

Boosting LHE scheme: Homomorphic properties ($\mathsf{Eval}_{\mathsf{B}}$)

Boosted-LHE multiplication function, level 1 (Mult_1)

- Inputs are ciphertexts $c_1 = [u_1, \beta_1]$ and $c_2 = [u_2, \beta_2]$ where $u_1, u_2 \in \mathcal{M}$ and $\beta_1, \beta_2 \in \widehat{C}$
- Compute $\alpha := \widehat{\mathsf{Enc}}(u_1 \cdot u_2) \boxplus (u_1 \boxdot \beta_2) \boxplus (u_2 \boxdot \beta_1)$
- Compute $\beta := \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix} = \begin{pmatrix} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{pmatrix}$
- Output is the ciphertext $c = [\alpha, \beta] = \mathsf{Enc}_{\mathsf{B}}(m_1 \cdot m_2)$

Correctness of Mult_1

Theorem

Assume that m_1, m_2 are messages from the message space $\mathcal{M}$ and b_1, b_2 are randomly picked numbers from $\mathcal{M}$. If $c_1 = [u_1, \beta_1] = \mathsf{Enc_B}(m_1)$, $c_2 = [u_2, \beta_2] = \mathsf{Enc_B}(m_2)$ and c is the output of $\mathsf{Mult}_1(c_1, c_2)$, then one can decrypt c and recover $m_1 \cdot m_2$.

Proof.

$$c = \mathsf{Mult}_1(c_1, c_2) = [\alpha, \beta]$$

where

$$\alpha := \widehat{\mathsf{Enc}}(u_1 \cdot u_2) \boxplus (u_1 \boxdot \beta_2) \boxplus (u_2 \boxdot \beta_1)$$
$$\beta := \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix} = \begin{pmatrix} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{pmatrix}$$

Correctness of Mult_1

Proof.

$$
\begin{aligned}
\alpha &= \widehat{\text{Enc}}((m_1 - b_1) \cdot (m_2 - b_2)) \boxplus ((m_1 - b_1) \boxdot \widehat{\text{Enc}}(b_2)) \\
&\boxplus ((m_2 - b_2) \boxdot \widehat{\text{Enc}}(b_1)) \\
&= \widehat{\text{Enc}}((m_1 - b_1) \cdot (m_2 - b_2)) \boxplus \widehat{\text{Enc}}((m_1 - b_1) \cdot b_2) \\
&\boxplus \widehat{\text{Enc}}((m_2 - b_2) \cdot b_1)) \\
&= \widehat{\text{Enc}}((m_1 - b_1) \cdot (m_2 - b_2) + ((m_1 - b_1) \cdot b_2) + ((m_2 - b_2) \cdot b_1)) \\
&= \widehat{\text{Enc}}(m_1 m_2 - b_1 m_2 - b_2 m_1 + b_1 b_2 + m_1 b_2 - b_1 b_2 + m_2 b_1 - b_1 b_2) \\
&= \widehat{\text{Enc}}(m_1 m_2 - b_1 b_2)
\end{aligned}
$$

and

$$\beta = (\widehat{\text{Enc}}(b_1), \widehat{\text{Enc}}(b_2))^T$$

Correctness of Mult_1

Proof.

Hence, one can recover $m_1 m_2$ as follows:

$$\begin{aligned}\widehat{\mathsf{Dec}}(\alpha) + \widehat{\mathsf{Dec}}(\beta_1) \cdot \widehat{\mathsf{Dec}}(\beta_2) &= \widehat{\mathsf{Dec}}(\alpha) + \widehat{\mathsf{Dec}}(\widehat{\mathsf{Enc}}(b_1)) \cdot \widehat{\mathsf{Dec}}(\widehat{\mathsf{Enc}}(b_2)) \\ &= (m_1 m_2 - b_1 b_2) + (b_1 \cdot b_2) \\ &= m_1 m_2 - b_1 b_2 + b_1 b_2 \\ &= m_1 m_2\end{aligned}$$

$\square$

Boosting LHE scheme: Homomorphic properties ($\mathsf{Eval_B}$)

Boosted-LHE addition function, level 2 (Add_2)

- Inputs are ciphertexts $c_1 = [\alpha_1, \beta_1] = \mathsf{Enc_B}(m_1)$ and $c_2 = [\alpha_2, \beta_2] = \mathsf{Enc_B}(m_2)$ where $m_1, m_2 \in \mathcal{M}$; $\alpha_1, \alpha_2 \in \widehat{C}$, $\beta_1 \in \widehat{C}^{2l_1}$ and $\beta_2 \in \widehat{C}^{2l_2}$; $\alpha_i = \widehat{\mathsf{Enc}}(m_i - b_i)$; $\beta_i := \begin{pmatrix} {\beta_{11}}^{(i)} & {\beta_{12}}^{(i)} & \dots & {\beta_{1l_i}}^{(i)} \\ {\beta_{21}}^{(i)} & {\beta_{22}}^{(i)} & \dots & {\beta_{2l_i}}^{(i)} \end{pmatrix}$ where ${\beta_{1k}}^{(i)} = \widehat{\mathsf{Enc}}(b_{1k}^{(i)})$, also ${\beta_{2k}}^{(i)} = \widehat{\mathsf{Enc}}(b_{2k}^{(i)})$ for some $b_{1k}^{(i)}, b_{2k}^{(i)} \in \mathcal{M}$ with $1 \le k \le l_i$ and $\sum_{k=1}^{l_i}[{b_{1,k}}^{(i)} \cdot {b_{2,k}}^{(i)}] = b_i$
- Compute $\alpha := \alpha_1 \boxplus \alpha_2$
- Compute $\beta := (\beta_1 || \beta_2) =$ $\begin{pmatrix} {\beta_{11}}^{(1)} & {\beta_{12}}^{(1)} & \dots & {\beta_{1l_1}}^{(1)} & {\beta_{11}}^{(2)} & {\beta_{12}}^{(2)} & \dots & {\beta_{1l_2}}^{(2)} \\ {\beta_{21}}^{(1)} & {\beta_{22}}^{(1)} & \dots & {\beta_{2l_1}}^{(1)} & {\beta_{21}}^{(2)} & {\beta_{22}}^{(2)} & \dots & {\beta_{2l_2}}^{(2)} \end{pmatrix}$
- Output is the ciphertext $c = [\alpha, \beta] = \mathsf{Enc_B}(m_1 + m_2)$

Boosted-LHE addition function, level 2 (Add_2)

Example: Inputs are ciphertexts $c_1 = [\alpha_1, \beta_1] = \mathrm{Enc}_{\mathrm{B}}(m_1)$ and $c_2 = [\alpha_2, \beta_2] = \mathrm{Enc}_{\mathrm{B}}(m_2)$ where $m_1, m_2 \in \mathcal{M}$; $\alpha_1, \alpha_2 \in \widehat{C}$, $\beta_1 \in \widehat{C}^{2l_1}$ and $\beta_2 \in \widehat{C}^{2l_2}$; $\alpha_i = \widehat{\mathrm{Enc}}(m_i - b_i)$;

Here $l_1 = l_2 = 1$. $\beta_1 := \begin{pmatrix} \beta_{11} \\ \beta_{21} \end{pmatrix}$ and $\beta_2 := \begin{pmatrix} \beta_{12} \\ \beta_{22} \end{pmatrix}$ where each $\beta_{jk} = \widehat{\mathrm{Enc}}(b_{jk})$

- Compute $\alpha := \alpha_1 \boxplus \alpha_2$
- Compute $\beta := (\beta_1 || \beta_2) = \begin{pmatrix} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{pmatrix}$
- Output is the ciphertext $c = [\alpha, \beta] = \mathrm{Enc}_{\mathrm{B}}(m_1 + m_2)$

Boosted-LHE addition function, level 2 (Add_2)

Observe that

$$\begin{aligned}\alpha = \alpha_1 \boxplus \alpha_2 &= \widehat{\text{Enc}}((m_1 - b_1) + (m_2 - b_2)) \\ &= \widehat{\text{Enc}}((m_1 + m_2) - (b_1 + b_2))\end{aligned}$$

where $b_1 + b_2 = [b_{11} \cdot b_{21}] + [b_{12} \cdot b_{22}]$.

Hence, one can recover $m_1 + m_2$ as follows:

$$\begin{aligned}\widehat{\text{Dec}}(\alpha) + \sum_{k=1}^{l_1+l_2} [\widehat{\text{Dec}}(\beta_{1k}) \cdot \widehat{\text{Dec}}(\beta_{2k})] &= \widehat{\text{Dec}}(\alpha) + [\widehat{\text{Dec}}(\beta_{11}) \cdot \widehat{\text{Dec}}(\beta_{21})] \\ &\quad + [\widehat{\text{Dec}}(\beta_{11}) \cdot \widehat{\text{Dec}}(\beta_{21})] \\ &= ((m_1 + m_2) - (b_1 + b_2)) + [b_{11} \cdot b_{21}] + [b_{12} \cdot b_{22}] \\ &= (m_1 + m_2) - (b_1 + b_2) + (b_1 + b_2) \\ &= m_1 + m_2.\end{aligned}$$

Correctness of Add_2

Theorem

If $c_i = [\alpha_i, \beta_i]$ *such that* $\alpha_i = \widehat{\mathrm{Enc}}(m_i - b_i)$ *for some* $b_i \in \mathcal{M}$,
$\beta_i := \begin{pmatrix} {\beta_{11}}^{(i)} & {\beta_{12}}^{(i)} & \dots & {\beta_{1l_i}}^{(i)} \\ {\beta_{21}}^{(i)} & {\beta_{22}}^{(i)} & \dots & {\beta_{2l_i}}^{(i)} \end{pmatrix}$ *where* ${\beta_{1k}}^{(i)} = \widehat{\mathrm{Enc}}(b_{1k}^{(i)})$, *also*
${\beta_{2k}}^{(i)} = \widehat{\mathrm{Enc}}(b_{2k}^{(i)})$ *for some* $b_{1k}^{(i)}, b_{2k}^{(i)} \in \mathcal{M}$ *with* $1 \le k \le l_i$ *and*
$\sum_{k=1}^{l_i} [{b_{1,k}}^{(i)} \cdot {b_{2,k}}^{(i)}] = b_i$, *then* c *can be computed (knowing* pk*) and given* c*, one can decrypt* c *and recover* $m_1 + m_2$ *(knowing* sk*).*

Proof.

$$c = \mathrm{Add}_2(c_1, c_2) = [\alpha, \beta]$$

where

□

Correctness of Add_2

Proof.

$$\alpha := \alpha_1 \boxplus \alpha_2$$

$$\begin{aligned}\beta &:= (\beta_1 || \beta_2) \\ &= \begin{pmatrix} {\beta_{11}}^{(1)} & {\beta_{12}}^{(1)} & \dots & {\beta_{1l_1}}^{(1)} & {\beta_{11}}^{(2)} & {\beta_{12}}^{(2)} & \dots & {\beta_{1l_2}}^{(2)} \\ {\beta_{21}}^{(1)} & {\beta_{22}}^{(1)} & \dots & {\beta_{2l_1}}^{(1)} & {\beta_{21}}^{(2)} & {\beta_{22}}^{(2)} & \dots & {\beta_{2l_2}}^{(2)} \end{pmatrix}\end{aligned}$$

Observe that

$$\begin{aligned}\alpha &= \alpha_1 \boxplus \alpha_2 \\ &= \widehat{\text{Enc}}((m_1 - b_1) + (m_2 - b_2)) \\ &= \widehat{\text{Enc}}((m_1 + m_2) - (b_1 + b_2))\end{aligned}$$

where $b_1 + b_2 = \sum_{k=1}^{l_1}[{b_{1,k}}^{(1)}.{b_{2,k}}^{(1)}] + \sum_{k=1}^{l_2}[{b_{1,k}}^{(2)}.{b_{2,k}}^{(2)}]$. □

Correctness of Add_2

Proof.

Hence, one can recover $m_1 + m_2$ as follows:

$$\widehat{\text{Dec}}(\alpha) + \sum_{k=1}^{l_1+l_2} [\widehat{\text{Dec}}(\beta_{1k}) \cdot \widehat{\text{Dec}}(\beta_{2k})]$$

$$= \widehat{\text{Dec}}(\alpha) + \sum_{k=1}^{l_1} [\widehat{\text{Dec}}({\beta_{1k}}^{(1)}) \cdot \widehat{\text{Dec}}({\beta_{2k}}^{(1)})] + \sum_{k=1}^{l_2} [\widehat{\text{Dec}}({\beta_{1k}}^{(2)}) \cdot \widehat{\text{Dec}}({\beta_{2k}}^{(2)})]$$

$$= ((m_1 + m_2) - (b_1 + b_2)) + \sum_{k=1}^{l_1} [{b_{1,k}}^{(1)} . {b_{2,k}}^{(1)}] + \sum_{k=1}^{l_2} [{b_{1,k}}^{(2)} . {b_{2,k}}^{(2)}]$$

$$= ((m_1 + m_2) - (b_1 + b_2)) + b_1 + b_2$$

$$= m_1 + m_2$$

$\square$

Boosting LHE encryption scheme: Construction

Decryption functions for the Boosted-LHE scheme ($\mathrm{Dec_B}$)

Boosted-LHE Decryption Level 1(Dec1)

- Inputs are ciphertext c, secret key $sk = a$
- Compute $m := u + \widehat{\mathrm{Dec}}(\beta)$
- Output the message m

Boosted-LHE Decryption Level 2(Dec2)

- Inputs are ciphertext c, secret key $sk = a$
- Compute $m := \widehat{\mathrm{Dec}}(\alpha) + \sum_{i=1}^{l}(\widehat{\mathrm{Dec}}(\beta_{1i}).\widehat{\mathrm{Dec}}(\beta_{2i}))$
- Output the message m

Correctness of Dec2

Theorem

If a level 2 ciphertext $c = [\alpha, \beta] \in C$ *is an encryption of* $m \in \mathcal{M}$*, then* $\mathrm{Dec2}(c) = m$*.*

Proof.

$$\mathrm{Dec2}([\widehat{\mathrm{Enc}}(m-b), \begin{pmatrix} \beta_{11} & \beta_{12} & \cdots & \beta_{1l} \\ \beta_{21} & \beta_{22} & \cdots & \beta_{2l} \end{pmatrix}])$$

where for each $\beta_{ik} = \widehat{\mathrm{Enc}}(b_{ik})$

$$\widehat{\mathrm{Dec}}(\alpha) + \sum_{i=1}^{l} [\widehat{\mathrm{Dec}}(\beta_{1i}).\widehat{\mathrm{Dec}}(\beta_{2i})]$$

□

Correctness of Dec2

Proof.

$$= m - b + \sum_{i=1}^{l} [\widehat{\mathrm{Dec}}(\widehat{\mathrm{Enc}}(b_{1i})).\widehat{\mathrm{Dec}}(\widehat{\mathrm{Enc}}(b_{2i}))]$$

$$= m - b + \sum_{i=1}^{l} [b_{1i} \cdot b_{2i}]$$

which finally yields $m - b + b$ and thus m. Hence we have, $\mathrm{Dec2}(c) = m$. □

Security of Boosted-LHE scheme

- Semantic security of HE_B depends on the semantic security of the scheme $\widehat{\mathsf{HE}}$.
- If $\widehat{\mathsf{HE}}$ is circuit private, then HE_B is also a leveled circuit private homomorphic encryption.

Efficiency of Boosted-LHE scheme

Table: **Efficiency of Boosted-LHE Encryption Scheme**

Functions	Operations
B-LHE Key generation	same as underlying LHE
B-LHE Encryption	1PRNG + 1S+ 1 LHE encryption
B-LHE Decryption	1A + 1 LHE decryption (for Dec1) and $(2l+1)$ LHE decryption + l M+ l A (for Dec2)
B-LHE Add_1	1A + 1 LHE A
B-LHE Mult_1	1M+ 2 LHE SM + 2 LHE A + 1 LHE encryption
B-LHE Add_2	1 LHE A
B-LHE SMult_1	1M+ 2 LHE SM
B-LHE SMult_2	$(l+1)$ LHE SM

Concluding remarks:

- We studied public key homomorphic encryption schemes: RSA, ElGamal, CGS.
- We studied a boosting technique for linearly homomorphic encryption schemes.
- We have full proofs of correctness.
- We implemented this boosting technique on the CGS scheme.
- We provided MAGMA source codes for CGS scheme and Boosted-CGS schemes.

Future works:

- Fully homomorphic enryption scheme, Gentry, in 2009.
- Full implementations:

  $\rightarrow$ to allow arbitrary multiplications.

  $\rightarrow$ to allow arbitrary additions on higher degree polynomials.
- Boosting multiplicative homomorphic schemes to allow additions.

References

Ronald Cramer, Rosario Gennaro and Berry Schoenmakers (1997)
A secure and Optimally Efficient Multi-authority election Scheme
EUROCRYPT, Lecture notes in Computer Science, Springer-Verlag 1233, 103–118.

Dario Catalano and Dario Fiore
Boosting Linearly-Homomorphic Encryption to Evaluate Degree-2 Functions on Encrypted Data

Thank you

Questions?