(Fully) Homomorphic Encryption and their Applications

Shanghai Jiao Tong University

May 14, 2019

Overview

A simple client-server HE scenario

Can we delegate the processing of data, without giving away access to it?

Solution: Homomorphic Encryption!
Application: Cloud Computing

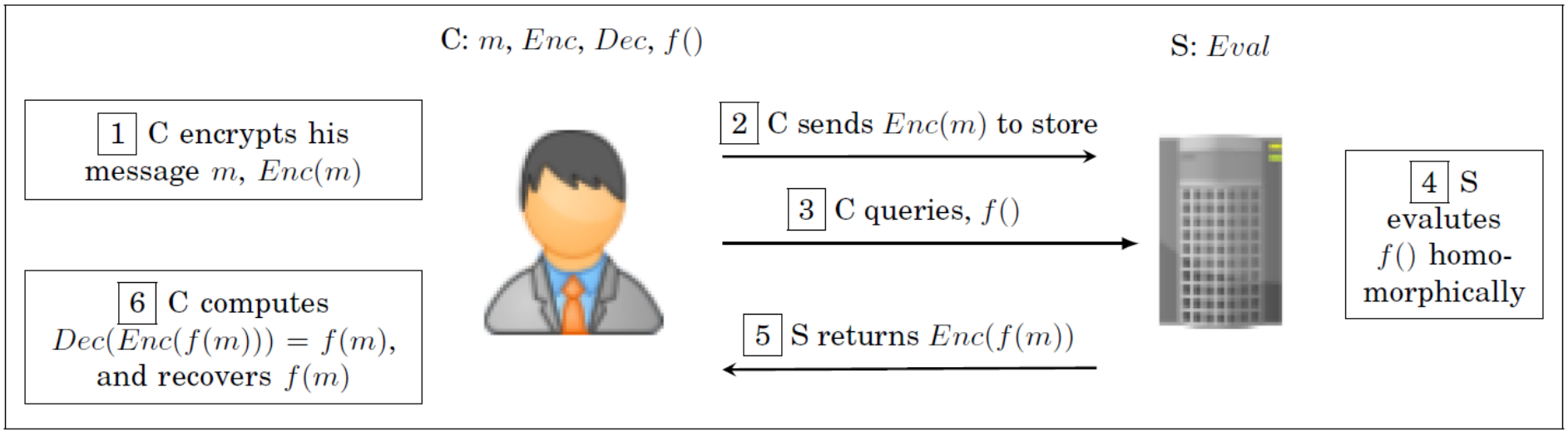
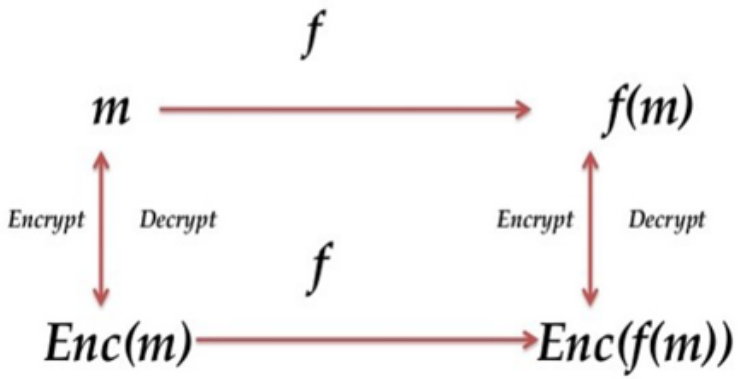


Figure: A simple client-server HE scenario [Abbas17]

(Fully) Homomorphic Encryption



A homomorphic encryption(HE) scheme allows computations on the ciphertext without knowing the secret key, meanwhile ensures that the decryption of the resulting ciphertext is exactly the same as the computations over the plaintext.

(Fully) Homomorphic Encryption Scheme

A homomorphic encryption scheme is a tuple (HE.KeyGen, HE.Enc, HE.Dec, HE.Eval) of probabilistic polynomial time (PPT) algorithms.

- $(sk, pk, evk) \leftarrow HE.KeyGen(1^\lambda)$
- $c \leftarrow HE.Enc(pk, m)$
- $m \leftarrow HE.Dec(sk, c)$
- $c_f \leftarrow HE.Eval(pk, f, evk, c_1, ..., c_l)$

FHE Correctness

Definition (Correctness)

We say that a homomorphic encryption correctly evaluates a circuit family $\mathcal{F}$ if for all $f \in \mathcal{F}$ and for all $m_1, ..., m_l \in \mathcal{M}$ it holds that if $(sk, pk) \leftarrow HE.KeyGen(1^\lambda)$, $c_i = HE.Enc_{pk}(m_i)$ for all i, and $c_f = HE.Eval_{pk,evk}(f, c_1, ..., c_l)$,

$$\Pr[HE.Dec_{sk}(c_f) \neq f(m_1, ..., m_l)] = negl(\lambda),$$

where the probability is taken over all the randomness in the experiment.

FHE Security

Definition (Security)

A homomorphic scheme is secure if any polynomial time adversary that first gets a properly generated pk, then specifies $m_0, m_1 \in \mathcal{M}$ and finally gets $HE.Enc_{pk}(m_b)$ for a random b, cannot guess the value of b with probability $> 1/2 + negl(\lambda)$.

In other words, $(pk, HE.Enc_{pk}(m_0)) \stackrel{c}{\approx} (pk, HE.Enc_{pk}(m_1))$ for any $m_0 \neq m_1$.

IND-CPA secure $\surd$
IND-CCA1 secure $\surd$ [CS98]
IND-CCA2 secure $\times$

Development of FHE

- Idea about privacy homomorphism was proposed [RAD78]
- Partially Homomorphic Encryption Schemes [RSA78] [Paillier99]
- 1^{st} generation FHE based on ideal lattice (bootstrapping) [Gen09b]
- 2^{nd} generation FHE based on RLWE (key/modulus switch) [BV11b][BGV12]
- 3^{rd} generation FHE based on LWE (approximate eigenvector) [GSW13]

Partially Homomorphic

[RSA78]

- KeyGen: $pk = (e, N), N = pq, gcd(e, \phi(n)) = 1$
- Enc: $Enc(m) = m^e \bmod N$
- Multiplicative Homomorphism:
  $Enc(m_1) \cdot Enc(m_2) = (m_1 m_2)^e \bmod N = Enc(m_1 m_2)$

[Paillier99]

- KeyGen: $pk = (n, g)$
- Enc: $Enc(m) = g^m \cdot r^n \bmod n^2$
- Additive Homomorphism:
  $Enc(m_1) \cdot Enc(m_2) = g^{m_1+m_2} \cdot (r_1 r_2)^n \bmod n^2 = Enc(m_1 + m_2)$

1^{st} generation FHE

[DGHV10] Integer version (Approximate GCD Problem)

- KeyGen: $sk = p$, p is a large odd integer
- Enc: randomness r, $|r| << p$, message $m \in \{0, 1\}$, $c = pq + 2r + m$
- Dec: $m = (c \bmod p) \bmod 2$
- Homomorphic Evaluation:
  $c^+ = (q_1 + q_2)p + 2(r_1 + r_2) + (m_1 + m_2)$
  $c^\times = (q_1 c_2 + q_2 c_1 - p q_1 q_2)p + 2(m_1 r_2 + r_1 m_2 + 2 r_1 r_2) + m_1 m_2$
- the above is a somewhat homomorphic encryption scheme

Bootstrapping

SWHE + Bootstrapping $\rightarrow$ (leveled) FHE

Bootstrapping: "refreshes" a ciphertext by running the decryption function on it homomorphically, resulting in a reduced noise.

$$c_1 \text{ under key } s_1 \text{ with noise } e_1 \overset{refresh}{\rightarrow} c_2 \text{ under key } s_2 \text{ with noise } e_2$$
$$\text{Dec}(c_1, s_1) = \text{Dec}(c_2, s_2) \text{ and } |e_2| < |e_1|$$

Theorem

Suppose $\mathcal{E}$ is a HE scheme

- *that can evaluate arithmetic circuits of depth d*
- *whose decryption algorithm is a circuit of depth $d-1$*

then we call $\mathcal{E}$ a "bootstrappable" HE scheme.

2^{nd} generation FHE

[BV11b][BGV12]

Definition (Leveled FHE)

A family of homomorphic encryption schemes $\{\mathcal{E}^L : L \in \mathbb{Z}^+\}$ is leveled fully homomorphic if

- they all use the same decryption circuit,
- $\mathcal{E}^L$ compactly evaluates all circuits of depth at most L,
- the computational complexity of $\mathcal{E}^L$'s algorithms is polynomial in the security parameter, L, and the size of the circuit.

Recall Ring-LWE

[Regev09]

Definition (Ring-LWE)

For security parameter λ, let $f(x) = x^d + 1$ where $d = d(\lambda)$ is a power of 2. Let $q = q(\lambda) \geq 2$ be an integer. Let $R = \mathbb{Z}[x]/(f(x))$ and let $R_q = R/qR$. Let $\chi = \chi(\lambda)$ be a distribution over R. The $RLWE_{d,q,\chi}$ problem is to distinguish the following two distributions:

- In the first distribution, one samples (a_i, b_i) uniformly from R_q^2 .
- In the second distribution, one first draws $s \leftarrow R_q$ uniformly and then samples $(a_i, b_i) \in R_q^2$ by sampling $a_i \leftarrow R_q$ uniformly, $e_i \leftarrow \chi$, and setting $b_i = a_i \cdot s + e_i$.

The $RLWE_{d,q,\chi}$ assumption is that the $RLWE_{d,q,\chi}$ problem is infeasible.

2^{nd} generation FHE

[BV11b][BGV12]

- KeyGen:$\mathbf{t} \leftarrow \chi^n$, $\mathbf{s} = (1, \mathbf{t}) \in R_q^{n+1}$, $\mathbf{C} \leftarrow R_q^{N \times n}$, $\mathbf{e} \leftarrow \chi^N$, $\mathbf{b} = \mathbf{Ct} + 2\mathbf{e}$, where $N = n\lceil logq \rceil$, set $\mathbf{A} = (\mathbf{b}| - \mathbf{C}) \in R_q^{N \times (n+1)}$. Observe that $\mathbf{As} = 2\mathbf{e}$.
- Enc: message $m \in R_2$, $\mathbf{m} = (m, 0, 0, ..., 0) \in R_2^{n+1}$, sample $\mathbf{r} \in R_2^N$ and output $\mathbf{c} = \mathbf{m} + \mathbf{A}^T\mathbf{r} \in R_q^{n+1}$
- Dec: $[[\langle \mathbf{c}, \mathbf{s} \rangle]_q]_2$, decryption works correctly because $[[\langle \mathbf{c}, \mathbf{s} \rangle]_q]_2 = [[(\mathbf{m}^T + \mathbf{r}^T\mathbf{A}) \cdot s]_q]_2 = [[m + 2\mathbf{r}^T\mathbf{e}]_q]_2 = [m + 2\mathbf{r}^T\mathbf{e}]_2 = m$

Definition (B-bounded distributions)

A distribution χ, supported over the integers, is called B-bounded if

$$Pr_{e \leftarrow \chi}[|e| > B] = negl(n)$$

2^{nd} generation FHE

[BV11b][BGV12]

- $[\langle \mathbf{c}_1, \mathbf{s}\rangle]_q = m_1 + 2e_1, [\langle \mathbf{c}_2, \mathbf{s}\rangle]_q = m_2 + 2e_2$
- Fact. $\langle \mathbf{c}_1 \otimes \mathbf{c}_2, \mathbf{s} \otimes \mathbf{s}\rangle = \langle \mathbf{c}_1, \mathbf{s}\rangle \cdot \langle \mathbf{c}_2, \mathbf{s}\rangle$
- Homomorphic Addition:
  $\mathbf{c}^+ = \mathbf{c}_1 + \mathbf{c}_2 = \mathbf{A}^T(\mathbf{r}_1 + \mathbf{r}_2) + (\mathbf{m}_1 + \mathbf{m}_2)$,
  $[[\langle \mathbf{c}^+, \mathbf{s}\rangle]_q]_2 = [2(e_1 + e_2) + (m_1 + m_2)]_2 = m_1 + m_2$
- Homomorphic Multiplication:
  tensor product $\mathbf{c}_3^\times = \mathbf{c}_1 \otimes \mathbf{c}_2$, $\mathbf{s}^\times = \mathbf{s} \otimes \mathbf{s}$,
  $[[\langle \mathbf{c}_3^\times, \mathbf{s}^\times\rangle]_q]_2 = [[(2e_1 + m_1)(2e_2 + m_2)]_q]_2 =$
  $[2(2e_1e_2 + m_1e_2 + m_2e_1) + m_1m_2]_2 = m_1m_2$

2^{nd} generation FHE (Key Switch)

[BV11b][BGV12]

- The dimension of the resulting multiplication ciphertext $\mathbf{c}_3^{\times}$ is $(n+1)^2$ instead of the original ciphertext dimension $(n+1)$.
- Question: How can we avoid ciphertext expansion?
- Solution: Key Switch!
- Convert $(n+1)^2$-dimension $\mathbf{c}_3$ under key $\mathbf{s}_3$ to $(n+1)$-dimension $\mathbf{c}_3'$ under key $\mathbf{s}_3'$ while preserving correctness.

Useful Subroutines (Key Switch)

[BV11b][BGV12]

- BitDecomp: $\mathbf{x} \in R_q^n, BitDecomp(\mathbf{x}) = (\mathbf{x}_0, ..., \mathbf{x}_{l-1}) \in R_2^{nl}$ such that $\mathbf{x} = \Sigma_{i=0}^{l-1} 2^i \mathbf{x}_i$ where $l = \lfloor logq \rfloor + 1$
- Powerof2: $\mathbf{y} \in R_q^n, Powerof2(\mathbf{y}) = (\mathbf{y}, 2\mathbf{y}, ..., 2^{l-1} \cdot \mathbf{y}) \in R_q^{nl}$

Lemma

$\langle BitDecomp(\mathbf{x}), Powerof2(\mathbf{y}) \rangle = \langle \mathbf{x}, \mathbf{y} \rangle$

Proof:

$$\begin{aligned}
\langle BitDecomp(\mathbf{x}), Powerof2(\mathbf{y}) \rangle &= \Sigma_{i=0}^{l-1} \langle \mathbf{x}_i, 2^i \cdot \mathbf{y} \rangle \\
&= \Sigma_{i=0}^{l-1} \langle 2^i \cdot \mathbf{x}_i, \mathbf{y} \rangle \\
&= \langle \Sigma_{i=0}^{l-1} 2^i \cdot \mathbf{x}_i, \mathbf{y} \rangle \\
&= \langle \mathbf{x}, \mathbf{y} \rangle
\end{aligned}$$

2^{nd} generation FHE (Key Switch)

[BV11b][BGV12]

- SwitchKeyGen($\mathbf{s}_1, \mathbf{s}_2, n_1, n_2, q$): $\mathbf{t}_2 \leftarrow R_q^{n_2}$, $\mathbf{s}_2 = (1, \mathbf{t}_2) \in R_q^{n_2+1}$, $\mathbf{C} \leftarrow R_q^{N \times n_2}$, $\mathbf{e}_2 \leftarrow \chi^N$, $\mathbf{b} = \mathbf{C}\mathbf{t}_2 + 2\mathbf{e}_2$, where $N = n_1\lceil logq \rceil$,set $\mathbf{A} = (\mathbf{b}| - \mathbf{C}) \in R_q^{N \times (n_2+1)}$. Observe that $\mathbf{A}\mathbf{s}_2 = 2\mathbf{e}_2 \in R_q^N$, add $powerof2(\mathbf{s}_1)$ to $\mathbf{A}$'s first column to get matrix $\tau_{\mathbf{s}_1 \to \mathbf{s}_2} = \mathbf{B}$.
- SwitchKey($\tau_{\mathbf{s}_1 \to \mathbf{s}_2}, \mathbf{c}_1, n_1, n_2, q$): For $\mathbf{c}_1 \in R_2^{n_1+1}$, the algorithm output $\mathbf{c}_2 = BitDecomp(\mathbf{c}_1)^T\mathbf{B} \in R_q^{n_2+1}$

$$
\begin{aligned}
\langle \mathbf{c}_2, \mathbf{s}_2 \rangle &= BitDecomp(\mathbf{c}_1)^T \cdot \mathbf{B} \cdot \mathbf{s}_2 \\
&= BitDecomp(\mathbf{c}_1)^T \cdot (2\mathbf{e}_2 + Powerof2(\mathbf{s}_1)) \\
&= 2\langle BitDecomp(\mathbf{c}_1), \mathbf{e}_2 \rangle + \langle BitDecomp(\mathbf{c}_1), Powerof2(\mathbf{s}_1) \rangle \\
&= 2\langle BitDecomp(\mathbf{c}_1), \mathbf{e}_2 \rangle + \langle \mathbf{c}_1, \mathbf{s}_1 \rangle \\
&= 2e + m_1 m_2
\end{aligned}
$$

2^{nd} generation FHE (Modulus Switch)

[BV11b][BGV12]

- The noise of the resulting ciphertext after multiplication is increasing dramatically.
- Question: How can we manage the noise amplification?
- Solution: Modulus Switch!
- Modulus Switch can transform a ciphertext $\mathbf{c}$ modulo q into a different ciphertext $\mathbf{c}'$ modulo p while preserving correctness, namely,

$$[\langle \mathbf{c}', \mathbf{s} \rangle]_p = [\langle \mathbf{c}, \mathbf{s} \rangle]_q \bmod 2, \text{ and } ||[\langle \mathbf{c}', \mathbf{s} \rangle]_p|| < ||[\langle \mathbf{c}, \mathbf{s} \rangle]_q||$$

Noise Management (Modulus Switch)

Noise Amplification

Suppose $q \approx B^k$, fresh ciphertext with noise of magnitude B

Noise increasing:

$$B \xrightarrow{1} B^2 \xrightarrow{2} B^4 \xrightarrow{3} \cdots \xrightarrow{logk} B^k$$

Modulus decreasing:

$$q \xrightarrow{1} q \xrightarrow{2} q \xrightarrow{3} \cdots \xrightarrow{logk} q$$

Can do *logk* levels of multiplication

Choose a ladder of gradually decreasing moduli $\{q_i \approx q/B^i\}$ for $i < k$

Noise increasing:

$$B \xrightarrow{1} B^2 \xrightarrow{switch} B \xrightarrow{2} B^2 \xrightarrow{switch} B \xrightarrow{3} \cdots \xrightarrow{k} B^2 \xrightarrow{switch} B$$

Modulus decreasing:

$$q \xrightarrow{1} q/B \xrightarrow{2} q/B^2 \xrightarrow{3} \cdots \xrightarrow{k} q/B^k$$

Can do k levels of multiplication

2^{nd} generation FHE (Modulus Switch)

[BV11b][BGV12]

Definition (Scale)

For integer vector $\mathbf{x}$ and integers $q > p$, we define $\mathbf{x}' \leftarrow \text{Scale}(\mathbf{x}, q, p, r)$ to be the R-vector closest to $(p/q) \cdot \mathbf{x}$ that satisfies $\mathbf{x}' = \mathbf{x} \bmod r$.

Lemma

Let d be the dgree of the ring. Let $q > p > r$ be positive integers satisfying $q = p = 1 \bmod r$. Let $\mathbf{c} \in R^n$ and $\mathbf{c}' \leftarrow Scale(\mathbf{c}, q, p, r)$. Then for any $\mathbf{s} \in R^n$ with $||[\langle \mathbf{c}, \mathbf{s} \rangle]_q|| < q/2 - (q/p) \cdot \gamma_R \cdot (r/2) \cdot \sqrt{d} \cdot l_1^{(R)}(\mathbf{s})$, we have

- $[\langle \mathbf{c}', \mathbf{s} \rangle]_p = [\langle \mathbf{c}, \mathbf{s} \rangle]_q \bmod r$ *(Correctness)*
- $||[\langle \mathbf{c}', \mathbf{s} \rangle]_p|| < (p/q) \cdot ||[\langle \mathbf{c}, \mathbf{s} \rangle]_q|| + \gamma_R \cdot (r/2) \cdot \sqrt{d} \cdot l_1^{(R)}(\mathbf{s})$ *(Reduced Noise)*

where $l_1^{(R)}(\mathbf{s}) = \sum_i ||\mathbf{s}[i]||$ for $\mathbf{s} \in R^n$.

2^{nd} generation FHE (Modulus Switch)

[BV11b][BGV12]

Proof.

For some integer k, we have $[\langle \mathbf{c}, \mathbf{s} \rangle]_q = \langle \mathbf{c}, \mathbf{s} \rangle - kq$.

For the same k, let $e_p = \langle \mathbf{c}', \mathbf{s} \rangle - kp \in R$.

Note that $e_p = [\langle \mathbf{c}', \mathbf{s} \rangle]_p \bmod p$. We claim that $||e_p||$ is so small that $e_p = [\langle \mathbf{c}', \mathbf{s} \rangle]_p$. We have

$$
\begin{aligned}
||e_p|| &= || - kp + \langle (p/q) \cdot \mathbf{c}, \mathbf{s} \rangle + \langle \mathbf{c}' - (p/q) \cdot \mathbf{c}, \mathbf{s} \rangle || \\
&\leq || - kp + \langle (p/q) \cdot \mathbf{c}, \mathbf{s} \rangle || + || \langle \mathbf{c}' - (p/q) \cdot \mathbf{c}, \mathbf{s} \rangle || \\
&\leq (p/q) \cdot ||[\langle \mathbf{c}, \mathbf{s} \rangle]_q|| + \gamma_R \cdot \sum_{j=1}^{n} ||\mathbf{c}'[j] - (p/q) \cdot \mathbf{c}[j]|| \cdot ||\mathbf{s}[j]|| \\
&\leq (p/q) \cdot ||[\langle \mathbf{c}, \mathbf{s} \rangle]_q|| + \gamma_R \cdot (r/2) \cdot \sqrt{d} \cdot l_1^{(R)}(\mathbf{s}) \\
&< p/2
\end{aligned}
$$

Furthermore, we have $[\langle \mathbf{c}', \mathbf{s} \rangle]_p = e_p = \langle \mathbf{c}', \mathbf{s} \rangle - kp = \langle \mathbf{c}, \mathbf{s} \rangle - kq = [\langle \mathbf{c}, \mathbf{s} \rangle]_q \bmod r$

2^{nd} generation FHE

[BV11b][BGV12]
SWHE + Key Switch + Modulus Switch $\rightarrow$ (leveled) FHE

- Key Switch: reduce the dimension of the ciphertext, transform a ciphertext $\mathbf{c}_1$ to another ciphertext $\mathbf{c}_2$ s.t. $m = Dec(\mathbf{c}_1, \mathbf{s}_1) = Dec(\mathbf{c}_2, \mathbf{s}_2)$.
- Modulus Switch: keep the noise level essentially constant, sacrifice modulus size and sacrifice the remaining homomorphic capacity.

(leveled) FHE + Circular Security $\rightarrow$ (pure) FHE

- Circular Security: It is "safe" to encrypt the leveled FHE secret key under its own public key.

2^{nd} generation FHE (Optimization)

[BV11b][BGV12]

- Batching: By packing multiple plaintexts into each ciphertext, allowing to evaluate a function f homomorphically in parallel on multiple blocks of encrypted data. (Chinese Remainder Theorem)
  - single ciphertext operation
  - many component-wise plaintext operation
- Batching the bootstrapping function: By running the decryption function homomorphically on multiple ciphertexts that need to be refreshed.
  - for circuits of large width

The above methods help to reduce the per-gate computation overhead.

3^{rd} generation FHE

[GSW13]

- KeyGen: $\mathbf{s} \leftarrow \mathbb{Z}_q^{n-1}$, private key $\mathbf{t} = (1, -\mathbf{s})$, $\mathbf{A} \leftarrow \mathbb{Z}_q^{(n-1)\times nl}$,$\mathbf{e} \leftarrow \chi^{nl}$, public key $\mathbf{P} = \begin{pmatrix} \mathbf{s}^T\mathbf{A}+\mathbf{e} \\ \mathbf{A} \end{pmatrix} \in \mathbb{Z}_q^{n\times nl}$
- Enc: $\mathbf{R} \leftarrow \{0,1\}^{nl\times nl}$, ciphertext $\mathbf{C} = \mathbf{PR} + m\mathbf{G} \in \mathbb{Z}_q^{n\times nl}$
- Dec: $\mathbf{tC} = m\mathbf{tG} + \mathbf{e}'$
- Homomorphic Evaluation:
  $\mathbf{t}(\mathbf{C}_1 + \mathbf{C}_2) = (m_1 + m_2)\mathbf{tG} + (\mathbf{e}_1 + \mathbf{e}_2)$
  $\mathbf{t}(\mathbf{C}_1 \cdot \mathbf{G}^{-1}(\mathbf{C}_2)) = (m_1\mathbf{tG} + \mathbf{e}_1) \cdot \mathbf{G}^{-1}(\mathbf{C}_2) = m_1 m_2\mathbf{tG} + (\mathbf{e}_1\mathbf{G}^{-1}(\mathbf{C}_2) + m_1\mathbf{e}_2)$
- $\mathbf{G} = \begin{pmatrix} g & & \\ & \ddots & \\ & & g \end{pmatrix}$ where $g = (1, 2, 4, ..., 2^l)^T$, note that for any $\mathbf{C} \in \mathbb{Z}_q^{n\times nl}$, we have $\mathbf{GG}^{-1}(\mathbf{C}) = \mathbf{C}$.

Applications of FHE

Domain	Genomics	Health	National Security	Education	Social Security	Business Analytics	Cloud
Sample Topics	*GWAS*	*billing and reporting*	*smart grid*	*school dropouts*	*credit history*	*prediction*	*storage, sharing*
Data Owner	*medical institutions*	*clinics and hospitals*	*nodes and network*	*schools, welfare*	*government*	*business owners*	*clients*
Why HE?	*HIPAA*	*cyber insurance*	*privacy*	*FERPA*	*cyber crimes*	*data are valuable*	*untrusted server*
Who pays?	*health insurance*	*hospital*	*energy company*	*DoE*	*government*	*business owners*	*clients*

Figure: Business Models and Application Domains [Tutorial]

Other Secure Computing Approaches

	HE	MPC	SGX
Performance	Compute-bound	Network-bound	
Privacy	Encryption	Encryption / Non-collusion	Trusted Hardware
Non-interactive	✔	✘	✔
Cryptographic security	✔	✔	✘ (known attacks)

Figure: How HE is different from MPC and SGX [Tutorial]

Library Matrix

Library/Scheme	FHEW	TFHE	BGV	BFV	CKKS
cuFHE		✔			
FHEW	✔				
FV-NFLlib				✔	
HEAAN					✔
HElib			✔		(✔)
PALISADE			✔	✔	(✔)
SEAL				✔	✔
TFHE(-Chimera)	✔	✔		(✔)	(✔)

Figure: Library Matrix [Tutorial]

References I

Abbas A , Hidayet A , Selcuk U A , et al (2017)

A Survey on Homomorphic Encryption Schemes: Theory and Implementation

Acm Computing Surveys, 51(4):1-35.

Ronald Cramer and Victor Shoup (1998)

A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack.

Advances in Cryptology - CRYPTO '98, 13-25.

Ron Rivest, Leonard Adleman, and Michael L. Dertouzos (1978)

On data banks and privacy homomorphisms.

Foundations of Secure Computation, 169-180, 1978.

Ronald L Rivest, Adi Shamir, and Len Adleman (1978)

A method for obtaining digital signatures and public-key cryptosystems.

Commun. ACM, 21, 2 (1978), 120-126.

Pascal Paillier (1999)

Public-key cryptosystems based on composite degree residuosity classes.

In Advances in cryptology-EUROCRYPT 99, 223-238.

References II

Craig Gentry (2009)

Fully homomorphic encryption using ideal lattices.

STOC, 169-178.

Zvika Brakerski and Vinod Vaikuntanathan (2011)

Efficient fully homomorphic encryption from (standard) LWE.

In Advances in Cryptology-CRYPTO, 505-524.

Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan (2012)

(Leveled) fully homomorphic encryption without bootstrapping.

In Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, 309-325.

Craig Gentry, Amit Sahai, and Brent Waters (2013)

Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based.

Advances in Cryptology-CRYPTO, 75-92.

Marten Van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan (2010)

Fully homomorphic encryption over the integers.

In Advances in cryptology-EUROCRYPT , 24-43..

References III

Homomorphic Encryption Standardization

http://homomorphicencryption.org/introduction/

Building Applications with Homomorphic Encryption

http://homomorphicencryption.org/wp-content/uploads/2018/10/CCS-HE-Tutorial-Slides.pdf

Oded Regev(2009)

On lattices, learning with errors, random linear codes, and cryptography

Journal of the ACM, 2009 pp. 56(6):1-40.

The End