Homomorphic Encryption Supporting Logical Operations

Yi-Fan Tseng
Department of Computer Science and Engineering, National Sun Yat-sen University
Kaohsiung, Taiwan
yftseng1989@gmail.com

Chun-I Fan*
Department of Computer Science and Engineering, National Sun Yat-sen University
Kaohsiung, Taiwan
cifan@mail.cse.nsysu.edu.tw

Ting-Chuan Kung
Department of Computer Science and Engineering, National Sun Yat-sen University
Kaohsiung, Taiwan
show_0929@hotmail.com

Jheng-Jia Huang
Department of Computer Science and Engineering,
National Sun Yat-sen University
Kaohsiung, Taiwan
jhengjia.huang@gmail.com

Hsin-Nan Kuo
Department of Computer Science and Engineering, National Sun Yat-sen University
Kaohsiung, Taiwan
bluedunk@gmail.com

ABSTRACT

Homomorphic encryption is a form of encryption that allows computations to be carried out on ciphertext and generate an encrypted result which, when decrypted, matches the result of operations performed on the plaintexts. The feature of homomorphic encryption is used in modern communication system architectures and cryptosystems. In view of the previous works, most of homomorphic encryptions support additive or multiplicative homomorphism. There is few homomorphic encryption schemes tailored for logical operations. In this paper, we propose a homomorphic encryption scheme that supports logical operations. Additionally, our proposed scheme can be applied to 2-DNF and k-CNF. Furthermore, the security of the proposed scheme is based on the subgroup decision assumption.

CCS Concepts

• Security and privacy → Public key encryption • Security and privacy→ Mathematical foundations of cryptography

Keywords

Homomorphic encryption; logical operations; composite order pairing; generic group model; disjunctive normal form

1. INTRODUCTION

Homomorphic encryption [1] is a form of encryption that allows computations to be carried out on ciphertexts and generate an encrypted result which, when decrypted, matches the result of operations performed on the plaintexts. For instance, one person could multiply two encrypted messages and then another person could decrypt the result, without either of them being able to calculate the content of the individual messages, as shown in Figure 1. One of the homomorphic encryption advantage is that the users' data privacy could be protected in cloud computing. Therefore, many papers discuss the homomorphic encryption applications, such as electronic voting [2][3][4][5], computational private information retrieval (PIR) [6] and private matching [7]. Many cryptosystems with homomorphic properties have been around for quite awhile, such as RSA [8], ElGamal [9], and Paillier [10]. The homomorphic property of RSA and ElGamal cryptosystem is multiplicative homomorphism. Furthermore, Paillier cryptosystem is additive homomorphism.

ICTCE 2017, October 22–24, 2017, Osaka, Japan
© 2017 Association for Computing Machinery.
ACM ISBN 978-1-4503-5315-1/17/10…$15.00
https://doi.org/10.1145/3145777.3145789

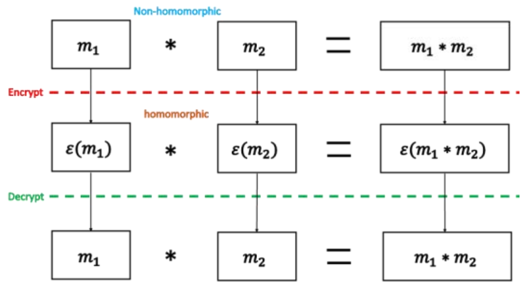

Figure 1. The property of homomorphic encryption

Many papers discuss the homomorphic encryption [11][12][13], but, most of homomorphic encryptions support additive or multiplicative homomorphism. There is few homomorphic encryption schemes tailored for logical operation homomorphism, such as [14]. As a result, we propose a homomorphic encryption scheme supporting logical operations. We construct the proposed scheme based on composite order pairing.

1.1 Contributions

We present a homomorphic encryption scheme supporting logical operations. Our proposed scheme can support AND and OR operation homomorphism. The proposed scheme can be applied to 2-DNF and k-CNF.

2. COMPOSITE ORDER BILINEAR GROUPS

Composite order bilinear groups were first introduced in [14]. We define them by using a group generation algorithm $\mathcal{G}$, which takes a security parameter λ as input and outputs bilinear groups $\mathbb{G}$ and $\mathbb{G}_T$ of order N. There are properties of composite order pairing as follows:

Definition 2.1. Let $\mathbb{G}$ and $\mathbb{G}_T$ be two cyclic groups of the order $N = pq$, p and q be distinct primes. A bilinear pairing is a map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ with following properties:

1. Bilinearity: $\forall g_1, g_2 \in \mathbb{G}, \forall a, b \in \mathbb{Z}_N, e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.
2. Non-degeneracy: $\exists g \in \mathbb{G}$ such that $e(g, g)$ has order N in $\mathbb{G}_T$.

Definition 2.2 (Canceling).
Let $\mathbb{G}_p$ and $\mathbb{G}_q$ denote the subgroups of $\mathbb{G}$ of order p and q. If g is a generator of $\mathbb{G}$, then g^q and g^p are generators of $\mathbb{G}_p$ and $\mathbb{G}_q$, respectively. For all elements $h_p \in \mathbb{G}_p$ and $h_q \in \mathbb{G}_q$, we have $e(h_p, h_q) = 1$, because $h_p = (g^q)^a, h_q = (g^p)^b$ for some $a, b \in \mathbb{Z}_N$ and $e(h_p, h_q) = e(g^{qa}, g^{pb}) = e(g^a, g^b)^{pq} = 1$.

This property will be a principal tool in our constructions.

3. OUR CONSTRUCTION

In this section, we present a homomorphic encryption scheme tailored for logical operations. The proposed scheme consists of three algorithms: $KeyGen, Encrypt, Decrypt$. The notations used in the proposed scheme are defined in Table 1.

Table 1. The notations

Notation	Meaning
$\mathbb{G}$	a cyclic group of order N
$\mathbb{G}_T$	a cyclic group of order N
λ	a security parameter
e	a bilinear mapping; $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$
p, q	distinct large primes
$\mathbb{G}_p$	a subgroup of $\mathbb{G}$ of prime order p
$\mathbb{G}_q$	a subgroup of $\mathbb{G}$ of prime order q
PK	public key
SK	secret key
M	message
C	ciphertext

3.1 The Proposed Scheme

$\mathbb{G}_p$ and $\mathbb{G}_q$ are two subgroups of prime order p and q. Let g_p be a generator of $\mathbb{G}_p$, g_q be a generator of $\mathbb{G}_q$ and a pairing $e: \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. The details of our scheme are described as follows.

- $KeyGen(\lambda)$:
  To generate the public keys and secret key, one performs the following steps:
  [1] Run group generation algorithm $\mathcal{G}(\lambda)$ to obtain a tuple $(p, q, \mathbb{G}, \mathbb{G}_T, e)$, $|\mathbb{G}| = |\mathbb{G}_T| = N = pq$, $\mathbb{G} = \mathbb{G}_p \times \mathbb{G}_q$.
  [2] Choose $Q \in_R \mathbb{G}_q$, and set $PK_0 = g_q$, $PK_1 = g_p Q$.
  [3] Set the public key $PK = (PK_0, PK_1)$, and set the secret key $SK = q$.
- $Encrypt(PK, M)$:
  To encrypt a bit M, one does the following steps:
  (1) Choose $r \in_R \mathbb{Z}_N$.
  (2) The message M is 0 or 1. There are two cases as follows:
  If M is 0, we use PK_0 to encrypt $M. \Rightarrow C = g_q^r$.
  If M is 1, we use PK_1 to encrypt $M. \Rightarrow C = (g_p Q)^r$.
  Note that the ciphertext C for $M = 0$ is an element in $\mathbb{G}_q$, while the ciphertext for $M = 1$ is not.
- $Decrypt(C, SK)$:
  We use $SK = q$ to decrypt C.
  $$M = \begin{cases} 0, \text{if } C^q = 1_{\mathbb{G}} \\ 1, \text{if } C^q \neq 1_{\mathbb{G}}. \end{cases}$$
  Note that if the ciphertext C is an element in $\mathbb{G}_T$, we can decrypt it in the similar way.
  $$M = \begin{cases} 0, \text{if } C^q = 1_{\mathbb{G}_T} \\ 1, \text{if } C^q \neq 1_{\mathbb{G}_T}. \end{cases}$$

3.2 Correctness

The correctness is demonstrated as follows:

If C is generated with $M = 0$, then we have $C \in \mathbb{G}_q$ since $C^q = (g_q^r)^q = (g^{pr})^q = (g^r)^{pq} = 1_{\mathbb{G}}$. If C is generated with $M = 1$, then we have $C \notin \mathbb{G}_q$ since $C^q = (g_p Q)^{rq} = (g^q Q)^{rq} = (g^r)^{q^2} (Q)^{rq} \neq 1_{\mathbb{G}}$.

3.3 Homomorphism

We present that our proposed scheme has homomorphism property. For simplicity, we denote C_b as the ciphertext with message $b \in \{0, 1\}$.

- The OR Operation:
  When we multiply two ciphertexts together, we can achieve OR operation homomorphism. For example, a ciphertext C_0 multiplied by a ciphertext C_0' is also an element of the cyclic group $\mathbb{G}_q$. We demonstrate the homomorphism of the OR operation $(C_b C_{b'})^t = C_{b \vee b'}$ with $t \in_R \mathbb{Z}_N$ as follows.

  Let $C_0 = g_q^{r_0} \in \mathbb{G}_q, C_0' = g_q^{r_0'} \in \mathbb{G}_q, C_1 = (g_p Q)^{r_1} \notin \mathbb{G}_q, C_1' = (g_p Q)^{r_1'} \notin \mathbb{G}_q$, and $t \in_R \mathbb{Z}_N$.

$$\begin{cases} (C_0 \cdot C_0')^t = \left(g_q^{r_0 + r_0'}\right)^t & \in \mathbb{G}_q \\ (C_0 \cdot C_1')^t = \left(g_p^{r_1'} \cdot \left(g_q^{r_0} \cdot Q^{r_1'}\right)\right)^t & \notin \mathbb{G}_q \\ (C_1 \cdot C_0')^t = \left(g_p^{r_1} \cdot \left(g_q^{r_0'} \cdot Q^{r_1}\right)\right)^t & \notin \mathbb{G}_q \\ (C_1 \cdot C_1')^t = \left((g_p Q)^{r_1 + r_1'}\right)^t & \notin \mathbb{G}_q \end{cases}$$

Table 2. The or operation

C_b	$C_{b'}$	$(C_b C_{b'})^t$	$b \vee b'$
$\in \mathbb{G}_q$	$\in \mathbb{G}_q$	$\in \mathbb{G}_q$	0
$\in \mathbb{G}_q$	$\notin \mathbb{G}_q$	$\notin \mathbb{G}_q$	1
$\notin \mathbb{G}_q$	$\in \mathbb{G}_q$	$\notin \mathbb{G}_q$	1
$\notin \mathbb{G}_q$	$\notin \mathbb{G}_q$	$\notin \mathbb{G}_q$	1

- The AND Operation:

  Note that in the proposed scheme shown in Subsection 3.1, "$M = 0$" is equivalent to "$C \in \mathbb{G}_q$" and "$M = 1$" is equivalent to "$C \notin \mathbb{G}_q$". We can extend the concept from $\mathbb{G}$ to $\mathbb{G}_T$ through bilinear maps. Let $\mathbb{G}_{Tq}$ be the subgroup of $\mathbb{G}_T$ with order q. If a ciphertext $\hat{C} \in \mathbb{G}_{Tq}$, then it represents a ciphertext for $M = 0$; if $\hat{C} \notin \mathbb{G}_{Tq}$ then it is a cipheretxt for $M = 1$. We demonstrate the homomorphism of the AND operation $e(C_b, C_{b'})^t = \hat{C}_{b \wedge b'}$ with $t \in_R \mathbb{Z}_N$ as follows.

  Let $C_0 = g_q^{r_0} \in \mathbb{G}_q, C_0' = g_q^{r_0'} \in \mathbb{G}_q, C_1 = (g_p Q)^{r_1} \notin \mathbb{G}_q, C_1' = (g_p Q)^{r_1'} \notin \mathbb{G}_q$, and $t \in_R \mathbb{Z}_N$.

$$\begin{cases} e(C_0, C_0')^t = e(g_q, g_q)^{r_0 r_0' t} & \in \mathbb{G}_{Tq} \\ e(C_0, C_1')^t = e(g_q, g_p)^{r_0 r_1' t} e(g_q, Q)^{r_0 r_1' t} & \in \mathbb{G}_{Tq} \\ e(C_1, C_0')^t = e(g_q, g_p)^{r_1 r_0' t} e(g_q, Q)^{r_1 r_0' t} & \in \mathbb{G}_{Tq} \\ e(C_1, C_1')^t = e(g_p, g_p)^{r_1 r_1' t} e(Q, Q)^{r_1 r_1' t} & \notin \mathbb{G}_{Tq} \end{cases}$$

Table 3. The and operation

C_b	$C_{b'}$	$e(C_b, C_{b'})^t$	$b \wedge b'$
$\in \mathbb{G}_q$	$\in \mathbb{G}_q$	$\in \mathbb{G}_{T_q}$	0
$\in \mathbb{G}_q$	$\notin \mathbb{G}_q$	$\in \mathbb{G}_{T_q}$	0
$\notin \mathbb{G}_q$	$\in \mathbb{G}_q$	$\in \mathbb{G}_{T_q}$	0
$\notin \mathbb{G}_q$	$\notin \mathbb{G}_q$	$\notin \mathbb{G}_{T_q}$	1

3.4 Discussions

In this subsection, we discuss the applications and the properties for the proposed scheme. How to apply the proposed scheme to 2-DNF and k-CNF is shown below.

- 2-DNF:

  We show how to apply our scheme to 2-DNF formula by constructing a function evaluation protocol. We consider two parties where Alice holds a 2-DNF Boolean formula $\phi(x_1, \ldots, x_n)$ and Bob holds an assignment $a = (a_1, \ldots, a_n)$. Finally, Bob learns $\phi(a)$.

  Definition 3.1. A 2-DNF formula over the variables $x_1, \ldots, x_n$ is of the form $\bigvee_{i=1}^{k} (\ell_{i,1} \wedge \ell_{i,2})$ where $\ell_{i,1}, \ell_{i,2} \in \{x_1, \ldots, x_n, \overline{x_1}, \ldots, \overline{x_n}\}$.

2-DNF Protocol.

Input: Alice holds a 2-DNF formula $\phi(x_1, \ldots, x_n) = \bigvee_{i=1}^{k} (\ell_{i,1} \wedge \ell_{i,2})$ and Bob holds an assignment $a = (a_1, \ldots, a_n)$ with $a_j \in \{0, 1\}$ for $j = 1, \ldots, n$. Both parties' inputs include a security parameter λ.

(1) Bob performs the following:

(a) He runs $KeyGen(\lambda)$ to generate PK, SK, and sends PK to Alice.

(b) He computes $C_{a_j} = Encrypt(PK, a_j)$ for $j = 1, \ldots, n$, $C_{\overline{a_j}} = Encrypt(PK, \overline{a_j})$ for $j = 1, \ldots, n$ and sends $C_a = \{C_{a_1}, \ldots, C_{a_n}\}, C_{\bar{a}} = \{C_{\overline{a_1}}, \ldots, C_{\overline{a_n}}\}$ to Alice.

(2) Alice performs the following:

(a) She receives the cipheretxts $C_a, C_{\bar{a}}$ and computes an encryption of ϕ by using our proposed logical operations. For each $\ell_{i,1}, \ell_{i,2}$ in a clause $(\ell_{i,1} \wedge \ell_{i,2})$, she picks the corresponding ciphertexts from C_a or $C_{\bar{a}}$. Then she does the homomorphic AND operation in each clause to obtain ciphertext C_i' for $i = 1, \ldots, k$. Next, she does the homomorphic OR operations on all (C_i')'s, that is, $C_1' \cdot C_2' \cdot \ldots \cdot C_k'$.

(b) She sends the result, i.e. the encryption of $\phi(a)$, to Bob.

(3) Bob decrypts the result.

- k-CNF:

  We show how to apply our scheme to the computation of a k-CNF formula by constructing a function evaluation protocol. Our scheme is only for k-CNF with one AND gate. We also consider two parties where Alice holds a k-CNF Boolean formula $\phi(x_1, \ldots, x_n)$ and Bob holds an assignment $a = (a_1, \ldots, a_n)$. Finally, Bob learns $\phi(a)$.

  Definition 3.2. A k-CNF formula over the variables $x_1, \ldots, x_n$ is of the form $(\bigvee_{i=1}^{k_1} \ell_{1,i}) \wedge (\bigvee_{i=1}^{k_2} \ell_{2,i})$ where $\ell_{1,i}, \ell_{2,i} \in \{x_1, \ldots, x_n, \overline{x_1}, \ldots, \overline{x_n}\}$.

k-CNF Protocol.

The k-CNF protocol is similar to the 2-DNF protocol.

Input: Alice holds a k-CNF formula $\phi(x_1, \ldots, x_n) = (\bigvee_{i=1}^{k_1} \ell_{1,i}) \wedge (\bigvee_{i=1}^{k_2} \ell_{2,i})$ and Bob holds an assignment $a = (a_1, \ldots, a_n)$ with $a_j \in \{0, 1\}$ for $j = 1, \ldots, n$. Both parties' inputs include a security parameter λ.

(1) Bob performs the following:

(a) He runs $KeyGen(\lambda)$ to generate PK, SK, and sends PK to Alice.

(b) He computes $C_{a_j} = Encrypt(PK, a_j)$ for $j = 1, \ldots, n$, $C_{\overline{a_j}} = Encrypt(PK, \overline{a_j})$ for $j = 1, \ldots, n$ and sends $C_a = \{C_{a_1}, \ldots, C_{a_n}\}, C_{\bar{a}} = \{C_{\overline{a_1}}, \ldots, C_{\overline{a_n}}\}$ to Alice.

(2) Alice performs the following:

(a) She receives the cipheretxts $C_a, C_{\bar{a}}$ and computes an encryption of ϕ by using our proposed logical operations. For each $\ell_{i,1}, \ell_{i,2}$ in the two cluases $(\bigvee_{i=1}^{k_1} \ell_{1,i})$ and $(\bigvee_{i=1}^{k_2} \ell_{2,i})$, she picks the corresponding ciphertexts from C_a or $C_{\bar{a}}$. Then she does the homomorphic OR operations in two clauses to obtain ciphertext C_1' and C_2'; Next, she does the homomorphic AND operation on C_1' and C_2'.

(b) She sends the result, i.e. the encryption of $\phi(a)$, to Bob.

(3) Bob decrypts the result.

3.5 Security Analysis

As shown in [14], composite order groups enjoy the hardness of subgroup decision problems. More precisely, there is no efficient algorithm being able to distinguish an element of $\mathbb{G}_q$ (or $\mathbb{G}_{Tq}$, respectively) from that of $\mathbb{G} - \mathbb{G}_q$ (or $\mathbb{G}_T - \mathbb{G}_{Tq}$, respectively). Observe that, in our scheme, the ciphertext of 0 is an element of $\mathbb{G}_q$ (or $\mathbb{G}_{Tq}$, respectively), while the ciphertext of 1 is in $\mathbb{G} - \mathbb{G}_q$ (or $\mathbb{G}_T - \mathbb{G}_{Tq}$, respectively). Therefore, based on the hardness of subgroup decision problems, there is no adversary that can reveal the message from the ciphertext.

4. COMPARISON

In this section, we describe the properties of our homomorphic encryption scheme and show the comparison with Boneh *et al.*'s scheme [14]. Furthermore, we show the computation costs. The properties and performance comparisons are summarized in Table 4 and Table 5, respectively.

4.1 Properties Comparison

In this subsection, we compare the properties with Boneh *et al.*'s scheme [14]. The property comparison is shown in Table 4.

Table 4. Property comparison

	[14]	Ours
Homomorphism	Arithmetic/Logical	Logical
Supporting 2-DNF	Yes	Yes
Supporting k-CNF	Yes	Yes

4.2 Performance Comparison

In this subsection, we compare the computation costs with Boneh *et al.*'s scheme [14]. The computation costs shown in Table 5 and the computing notations are defined in Table 6.

Table 5. Performance comparison

	[14]	Ours
Encryption cost	$T_m + T_e$	T_e
Decryption cost	T_e	T_e
OR gate cost	$2T_m + T_e$	$T_m + T_e$
AND gate cost	$T_p + T_m + T_e$	$T_p + T_e$

Table 6. The notations

Notation	Meaning
T_m	the cost of a multiplication
T_e	the cost of an exponentiation
T_p	the cost of a pairing computation

5. CONCLUSION

In this paper, a homomorphic encryption scheme for logical operations has been proposed. First, the proposed scheme supports OR and AND operation homomorphism. Second, the proposed scheme can be applied to 2-DNF and k-CNF. In the future, we will try to extend the proposed scheme for NOT operation homomorphism and k-DNF.

6. ACKNOWLEDGEMENTS

This work was partially supported by the Ministry of Science and Technology of the Taiwan under grants MOST 105-2221-E-110-053-MY2, MOST 105-2923-E-110-001-MY3, and MOST 106-3114-E-110-001.

7. REFERENCES

[1] Yi, X., Paulet, R., and Bertino, E. 2014. Russell Paulet, and Elisa Bertino. *Homomorphic encryption and applications*. Springer. DOI= 10.1007/978-3-319-12229-8.

[2] Benaloh, J. D. C. 1987. Verifiable secret-ballot elections.

[3] Cohen, J. D., and Fischer, M. J. 1985. *A robust and verifiable cryptographically secure election scheme*. Yale University. Department of Computer Science.

[4] Cramer, R., Franklin, M., Schoenmakers, B., and Yung, M. 1996. Multiauthority secret-ballot elections with linear work. In *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 72–83. DOI= 10.1007/3-540-68339-9_7.

[5] Cramer, R., Gennaro, R., and Schoenmakers, B. 1997. A secure and optimally efficient multi-authority election scheme. *Transactions on Emerging Telecommunications Technologies*. 8, 5 (Sep. 1997), 481–490. DOI= 10.1002/ett.4460080506.

[6] Kushilevitz, E., and Ostrovsky, R. 1997. Replication is not needed: Single database, computationally-private information retrieval. In *Foundations of Computer Science, 1997. Proceedings., 38th Annual Symposium on*. IEEE, 364–373. DOI= 10.1109/SFCS.1997.646125.

[7] Freedman, M. J, Nissim, K., and Pinkas, B. 2004. Efficient private matching and set intersection. In *International conference on the theory and applications of cryptographic techniques*. Springer, 1–19. DOI= 10.1007/978-3-540-24676-3_1.

[8] Rivest, R. L, Shamir, A., and Adleman, L. 1978. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*. 21, 2 (Feb. 1978), 120–126. DOI= 10.1145/359340.359342.

[9] ElGamal, T. 1985. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*. 31, 4 (Jul. 1985), 469–472. DOI= 10.1109/TIT.1985.1057074.

[10] Paillier, P. 1999. Public-key cryptosystems based on composite degree residuosity classes. In *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 223–238. DOI= https://doi.org/10.1007/3-540-48910-X_16.

[11] Goldwasser, S., and Micali, S. 1982. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*. ACM, 365–377. DOI= 10.1145/800070.802212.

[12] Morris, L. 2013. Analysis of partially and fully homomorphic encryption. *Rochester Institute of Technology*. New York, 1–5.

[13] Naehrig, M., Lauter, K., and Vaikuntanathan, V. 2011. Can homomorphic encryption be practical?. In *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*. ACM, 113–124. DOI= 10.1145/2046660.2046682.

[14] Boneh, D., Goh, E.-J., and Nissim, K. 2005. Evaluating 2-dnf formulas on ciphertexts. In *Theory of Cryptography Conference*. Springer, 325–341. DOI= 10.1007/978-3-540-30576-7_18.