Open Access
2021, Vol. 18 No.2 (Suppl. June)
Baghdad Science Journal
P-ISSN: 2078-8665
E-ISSN: 2411-7986

DOI: http://dx.doi.org/10.21123/bsj.2021.18.2(Suppl.).0899

A Modified Symmetric Key Fully Homomorphic Encryption Scheme Based on Read-Muller Code

RatnaKumari Challa [1*] *VijayaKumari Gunta* [2]

[1] Rajiv Gandhi University of Knowledge Technology – Andhrapradesh, India.
[2] Jawaharlal Nehru Technological University Hyderabad, Telangana, India.
[*]Corresponding author: ratnamala3784@gmail.com [*], vijaykumri.gunta@gmail.com
[*]ORCID ID: https://orcid.org/0000-0001-5077-8513 [*]

Received 28/3/2021, Accepted 12/4/2021, Published 20/6/2021

Abstract:

Homomorphic encryption became popular and powerful cryptographic primitive for various cloud computing applications. In the recent decades several developments has been made. Few schemes based on coding theory have been proposed but none of them support unlimited operations with security. We propose a modified Reed-Muller Code based symmetric key fully homomorphic encryption to improve its security by using message expansion technique. Message expansion with prepended random fixed length string provides one-to-many mapping between message and codeword, thus one-to many mapping between plaintext and ciphertext. The proposed scheme supports both (MOD 2) additive and multiplication operations unlimitedly. We make an effort to prove the security of the scheme under indistinguishability under chosen-plaintext attack (IND-CPA) through a game-based security proof. The security proof gives a mathematical analysis and its complexity of hardness. Also, it presents security analysis against all the known attacks with respect to the message expansion and homomorphic operations.

Key words: Chosen Plaintext Attack, Homomorphic Encryption, Random Permutation, Reed-Muller Code, Sparse Subset Sum.

Introduction:

There has been a continuous shift towards cloud computing and it has provided as a means of public storage and related services (1, 2). At the same time, data stored in the public cloud are more vulnerable to unauthorized access as well as attacks. Adding security to the data using encryption will give problem to public access (3). Homomorphic encryption helps in providing public access and security on the data in a single step. Homomorphic encryption schemes satisfy an important property that, given two ciphertexts say $c_1 = Encrypt_k(m_1)$ and $c_2 = Encrypt_k(m_2)$ where m_1, m_2 are plaintexts and k is the key, one can compute $c = c_1 o\ c_2 = Encrypt_k(m_1 o\ m_2)$ for some operation o such that $Decrypt_k(c) = m_1 o\ m_2$. For example, the text book RSA encryption scheme is multiplicatively homomorphic (4). If o corresponds to only a *single* operation like addition or multiplication, such a scheme is called *partially homomorphic*. Several partially homomorphic encryption schemes were proposed and successfully used in the applications such as oblivious polynomial evaluation, electronic voting, multiparty computation, private information retrieval, Deep Learning systems, Big data systems, medical applications and so on (5-8). However, in order to perform arbitrary computations over the encrypted data so that the scheme is suitable for any application in general, it must support both addition and multiplication operations over the ciphertexts unlimitedly. Such an encryption scheme that supports unlimited additions and multiplications and thus, allows arbitrary computations over the encrypted data, is called as *Fully Homomorphic Encryption* (FHE) scheme. The problem of constructing an FHE scheme has been a dream of cryptographers, which was first theoretically solved in a pioneering work by Craig Gentry using an innovative construction method (9, 10).

Because of time complexities of Gentry's generic blueprint, most of the later work tried to bring Gentry's scheme close to practicality or used different security assumptions to create an FHE with practical time complexities (11-17). Even

though all of these have shown progressive improvements one over the other, none of them could qualify for practical implementation. The design of an FHE scheme with implementation model is still a challenge.

HE schemes based on coding theory are of interest because of the availability of alternative security assumptions in solving multivariate equations over a finite field and simplicity in decoding operation (18, 19) Because of simplicity of the linear mapping in decoding function, homomorphic operations can be supported by encryption schemes based on coding theory (18). Schemes based on Reed-Muller codes, McEliece codes and Goppa code based on McEliece have also not been homomorphic in nature (19-25). Hence, these schemes do not support computation of arbitrary functions over the encrypted data. Another McEliece code-based scheme supports homomorphic addition and does not support homomorphic multiplication operation (26).

Armknecht et al. (2011) presented a first code-based Somewhat Homomorphic Encryption (SHE) scheme using Reed-Muller codes. The computation complexities of this scheme stands at $O(n^2)$ and $O(n)$ for encryption and decryption respectively and at $O(n)$ for the homomorphic addition and multiplication operations. Anew Reed-Muller Code (RMC)-based FHE scheme, based on the scheme presented by Armknecht et al. (2011), has been proposed with a de-noising step which nullifies the error terms produced during each homomorphic multiplication (27). This is, in one way, the ciphertext refreshing or *post-processing* idea similar to bootstrapping step in the Gentry's blueprint (9).

The RMC-based FHE scheme (27) may have vulnerability owing to one-to-one mapping between message (plaintext) and codeword. The underlying codeword in the ciphertext at the fixed positions, as specified by the secret key, might be computed through Chosen Plaintext Attack (CPA). Present work is the modification to the RMC-based FHE scheme (27) by padding mechanism in order to give one-to-many mapping between message and codeword which will improve its security against CPA. The scheme is proved CPA secure and is thoroughly analyzed with respect to the changes and new techniques suggested to show that the proposed scheme is secure against all the known attacks.

The proposed scheme with its security proof and security analysis is presented in the subsequent sections of the paper. The proposed modified RMC-based symmetric key FHE scheme is detailed in the section Modified RMC-based symmetric key FHE Scheme. Game based security proof in mathematical expression is provided in section The proposition is CPA secure. Security analysis against the known attacks is discussed and work is concluded in last section.

Modified RMC-Based Symmetric Key FHE Scheme:

Notations

The symbols and their meaning used in this work are given as follows

1. F is an arbitrary finite field
2. Vectors have been denoted by small case bold letters e.g. v
3. $'+'$ denotes addition operation over GF(2) fields
4. $'.'$denotesmultiplication operation over GF(2) fields
5. An integer is denoted as a lower-case italic letter eg.x
6. $[n]$, integer set, contains values $\{1, 2, \dots, n\}$

Proposed Fully Homomorphic Encryption scheme

In the proposed scheme, Reed Muller encoding and decoding operations have been used. The decoding algorithm used here is similar to the decoding used in the RMC-based FHE scheme (27). In encryption step, the novel method is adopted to expand the message by prepending a random binary string of zeros and ones in order to generate a new (nondeterministic) codeword for same message if the same message encoded multiple times, which will provide one-many mapping between message and codeword.

Overview of the scheme

Like any other FHE scheme, *KeyGen*, *Encrypt*, *Decrypt*, and *Evaluate* algorithms have been used for the proposed scheme.

In the *Keygen* algorithm, given the security parameter$p$ $(<k)$, the RM parameters(r, m) are chosen, which determine the length of the codewords and the maximum length of the message that can be encoded. The actual plaintext chosen for encryption will be smaller than the maximum length$k$of the message that can be encoded by the RM(r,m). This enhances security. Precisely, the length of the actual plaintext to be encrypted is limited to the parameter$p$. Upon choosing the RM parameters(r, m), the length of the codeword$n$, dimension k, the generator matrix $GM_{r,m}$ for the code RM(r, m) and the length of the ciphertext$l$ are computed. A secret key$K$ is chosen as a random subset of the set of integers $\{1, 2, \dots, l\}$. The key K represents set of bit positions, random in nature, within a binary string of length$l$.It means $K$is set of

locationsat which each bit of the codewordare to be embedded during encryption. The parameters r, m, n, k, the generator matrixGM, and the key K all are kept secret.

The *Encrypt* algorithm takes the plaintext vector p of length$\leq p$, the generator matrix$GM_{r,m}$, maximum length of the message$k$, and the key$K$ as inputs and produces ciphertext vector$c$of length $l$as an output. Before encryption, the given plain text p is expanded to a message of length$k$ by prepending a zero vector0 to make the message to the length p, and then prepending a random bit vector$r$of appropriate lengthto p. For easy recovery of the plaintext p at decryption stage, it is recommended to choose$r$ of length$(k-p)$. In fact, appending the vector0is optional, which can be done only when the given plaintext $p$is short of the length p.

The generator matrix$GM_{r,m}$ can be used to transform the expanded plaintext message $m \in F_2^k$ into an$n$-bit codeword w**.** Then, the code word w is used to generate the ciphertext$c$. To achieve this, bits of w are embedded in$c$ (al-bit random vector) at locations specified by$K$.

Decrypt algorithm takes secretkey K and ciphertext c, it recovers codeword w from $c$by collecting the bits from the locations specified by K. Then the recovered $w$is decoded to produce the expanded plaintext message m from which the actual plaintext $p$is recovered by discarding the first $(k-p)$ bits followed by discarding the zero bits till the first non-zero bit is encountered.

The procedure, *H.Add* has been used to perform homomorphic addition operation and the procedure *H.Mul* has been used to perform homomorphic multiplication to construct the *Evaluate*algorithm. A homomorphic component wise $mod\ 2$ addition can be performed because of structure of ciphertexts. But same thing doesn't hold true for homomorphic component wise$mod\ 2$ multiplication. To overcome this problem, one more step of *de-noising* has been added to multiplication operation to nullify the error introduced as discussed in the RMC-based FHE scheme (27).

Algorithms

The algorithms describing the construction of the proposed scheme are presented as follows:

$KeyGen(1^p) \rightarrow (GM, K, l)$: Upon input of the parameter p,

1. Choose the r, mvalues such that $k \geq 2p$ and $n \geq 2k$ where $k = 1 + \binom{m}{1} + \binom{m}{2} + \cdots + \binom{m}{r}$ and $n = 2^m$
2. Construct generator matrix GM for $RM(r, m)$
$$GM = \begin{bmatrix} \frac{v_0}{v_1} \\ \hline \vdots \\ \hline v_{k-1} \end{bmatrix}$$
3. Choose $l \geq n^2$
4. Select a random subset $S \subset [l]$ of size n
5. Obtain a random permutation of S, and let the permuted set be the key, K
6. Output the GM, the key K and l.

$Encrypt(m, GM, K) \rightarrow c$ **:** This function takes plaintext $m = (a_0, a_1, \ldots, a_{p-1})$, the generator matrix GM and the key K will produce the ciphertext as output through following steps

1. If the size of m is less than p, concatenate m with a zero vector 0such that the size of $0||m$ is equal to p
2. Choose a random bit vector $r$of size $(k-p)$
3. Concatenate r and $m$to obtain the expanded plaintext, $p = (p_0, p_1, \ldots, p_{k-1})$
4. Compute the codeword vector, $w = p_0 v_0 \oplus p_1 v_1 \oplus \ldots \oplus p_{k-1} v_{k-1} = (x_0, x_1, \ldots, x_{n-1})$using the generator matrix $GM = \{v_0, v_1, \ldots, v_{k-1}\}$
5. Generate an l-bit random vector l
6. Replace the bits of l, as specified by K, with the bits of codeword to get the ciphertext c

$Decrypt(c, K) \rightarrow m$ **:**This function takes the ciphertext c and the key K to decipher the message m as follows

1. Retrieve the codeword, $w = (x_0, x_1, \ldots, x_{n-1})$ from c by collecting from the locations as specified by the key K
2. Compute the k-bit expanded plaintext message $p = (p_0, p_1, \ldots, p_{k-1})$ as follows

i) $p_0 = x_0$

ii) for $i = 1$ to $k-1$

$p_i = x_0 \oplus x_{2^{i-1}}$

3. Remove the $(k-p)$ most significant bits from $p$to obtain $0||m$
4. Remove the most significant zero bits from resulting bit vector if any in it.
5. Output the resulting plaintext $m = (a_0, a_1, \ldots, a_{p-1})$

$Evaluate(c_1, c_2, o) \rightarrow c_e$ **:**Given two ciphertexts $c_1$andc_2 corresponding to the messages m_1 and m_2 respectively, the homomorphic operation o, which

may be addition (*H.Add*) or multiplication (*H.Mul*), is performed as follows:

H.Add is $mod\ 2$ addition operation over the ciphertexts. It is denoted as $c_e = c_1 + c_2$. Decryption of c_e gives $m_1 + m_2$.

Similarly,*H.Mul* is $mod\ 2$ multiplication over the ciphertexts. Unlike addition (*H.Add*), mod 2 multiplication is not straightforward due to noise that appear in the resultant ciphertext $c_e = c_1.c_2$. But Multiplication with de-noising step as discussed in RMC-baseed FHE scheme (27) eliminates the additional noise generated during multiplication and decryption of resulting cipher produces $m_1.m_2$.

The Proposition is CPA Secure:

It may be noticed that the *key vector* is an instance of the sparse subset sum problem (9-11). However, the sum of the bit positions or any such aggregate corresponding to key positions is not being mentioned anywhere. So, this becomes a hidden sparse subset sum problem (28) for which no solution is known. So, the security of the scheme is totally dependent on the size and randomness of the ciphertext and the secret key permutation chosen, which involves the positions at which the codeword is to be embedded. Once the codeword is recovered, decoding is simple. In general, no explicit hard problem assumption is known for the security of a cryptosystem which involves embedding the plaintext bits in a randomly generated bit stream.

Similar studies have been performed and proved its security following game-based security proof (29-31) and the security of present work can also be proved on same lines as follows.

The IND-CPA game

For a probabilistic symmetric key algorithm, IND-CPA is defined by the following game between an adversary A and a challenger C:

A is assumed to be a probabilistic polynomial time (PPT) algorithm. That means, A must complete the game and output a *guess* within a polynomial number of steps (31). Let the parameters have their usual meaning as described in the scheme Notations. Let the proposed symmetric key encryption scheme be S= (*KeyGen*, *Encrypt*, *Decrypt*, *Evaluate*). Then the IND-CPA game, adapted from article (32), is defined as follows

Symmetric key CPA indistinguishability game: $SymK_{A,S}^{CPA}(p)$ $\mathrm{Sym}_{A,S}^{\mathrm{CPA}}\mathrm{SymK}_{A,S}^{\mathrm{CPA}}$

1. C generates a random secret key$K$ based on the security parameter $p$and retains K. In the proposed scheme, the size of the ciphertext$l$ and the key$K$ are dependent on the RM parameters(r, m).
2. A is given the security parameter 1^p and an oracle access to $Encrypt(.)$. It, then, performs a polynomial number of encryptions and other computations and chooses a pair of plaintext messagesm_0 , $m_1$such that,$|m_0| = |m_1| \leq p$ and sends them to C.
3. C chooses$b \leftarrow \{0,1\}$uniformly at random, and sends the *challenge* ciphertext$c = Encrypt(m_b, GM, K)$to A.
4. A continues to have oracle access to$Encrypt(.)$ and may perform a polynomial number of additional encryptions or computations and random guesses over the key$K$ the codeword w and the other parameters involved in the encryption to generate a guess for$b$, sayb'.
5. The adversary wins the game if$b' = b$, in case of which the output of the game is defined as 1, and 0 otherwise.

Definition 1(32)**.** A symmetric key encryption scheme, say S = (*KeyGen, Encrypt, Decrypt, Evaluate*) is IND-CPA secure or has *indistinguishable encryptions under chosen plaintext attack* if for all PPT adversaries A there exists a negligible function say$\varepsilon(.)$ such that,

$$Pr\ Pr\left[SymK_{A,S}^{CPA}(p) = 1\right] \leq \frac{1}{2} + \varepsilon(p)$$

That means, an encryption scheme is IND-CPA secure if every PPT adversary has only a ***negligible advantage*** over random guessing. The advantage of an adversary is said to be negligible if it wins the above game with the probability $\frac{1}{2} + \varepsilon(.)$, where$\varepsilon(p)$ is a negligible function in the security parameter$p$.

Definition 2 (32)**.** A function$f(n)$ is said to be negligible if for every$c \in N$, there exists an integern_0 such that,$f(n) \leq \frac{1}{n^c}$for all $n \geq n_0$.

Now, let$l$ $R \leftarrow \{0,1\}^l$ is a random$l$-bit string. We claim that, under the assumption of ***random embedding*** of the permuted$w$ in$l$, no PPT adversary A can learn *any information* about the plaintext$m$or codeword$w$(beyond what it could guess at random) except with negligible probability. This is proved in the following theorem.

Theorem 1. Let the proposed symmetric key encryption scheme, S = (*KeyGen, Encrypt, Decrypt, Evaluate*). Let$p$be the security parameter and the other parameters$r,\ m,\ k,\ l,\ n$ are

chosen/computed as defined. An adversary A that breaks the IND-CPA security of the proposed scheme will have an advantage $\varepsilon(p)$, where $\varepsilon(p)$ is negligible.

Proof. Given the two plaintexts m_0, m_1, it is evident that the probability of choosing the one uniformly at random by the challenger is $\frac{1}{2}$. We analyse the probability the adversary A will have over and above this random guess. Given a ciphertext c of length l, A first tries to obtain the length of the codeword n. The only way it can do this is by assuming that $l = n^2$, the minimum length of the ciphertext, because, we keep the parameters r, m as secret and $l \geq n^2$. Based on the value of n, A selects a subset of size n from $[\,l\,]$ which he can retrieve the codeword. This probability is $\frac{1}{(l\ n)}$. As another way, A may directly guess a codeword w from the set of all possible strings of length n. In fact, only few of these strings, i.e., $2^k << 2^n$ are the codewords. Suppose that, A is able to compute k. Even then, the permuted codeword may not belong to the set of codewords of dimension k and the problem boils down to choosing a string from the set of 2^n strings. Clearly, this probability is $\frac{1}{2^n}$.

Upon choosing the permuted embedded codeword, A has to compute the exact permutation of the n-bit string chosen to obtain the actual embedded codeword. However, the number of permutations depends on the pattern of the strings. The codewords like all 0's and all 1's will make no sense with respect to the permutation. So, we assume that the permutation can be guessed with a probability of 1 in such cases. This makes the whole probability of guessing a codeword and its permutation in worst case as $\left(\frac{1}{2^n}\right).1 = \frac{1}{2^n}$.

So, the overall probability of A to generate a ciphertext hopefully closer to the one returned by the challenger is $\frac{1}{(l\ n)} + \frac{1}{2^n}$, which is nothing but the advantage $\varepsilon(p)$ it has over the random guess $1/2$.

For simplicity, let us take $k = 2p$ and $n = 2k$ as mentioned in the *KeyGen* algorithm. This gives $n = 4p$ and $l = 16\,p^2$. Substituting these values in $\frac{1}{(l\ n)} + \frac{1}{2^n}$, we get $\left(\frac{(16p^2-4p)!(4p)!}{16p^2\,!}\right) + \frac{1}{16^p}$. Thus the advantage of the adversary $\varepsilon(p) = \left(\frac{(16p^2-4p)!(4p)!}{16p^2\,!}\right) + \frac{1}{16^p}$. In this function $\varepsilon(p)$, the exponential component $\frac{1}{16^p}$ is clearly negligible (32). The Lemma 1 below, shows that the factorial based component, which is inverse a combination function is also negligible for sufficiently large values of p. Hence, from the Proposition1 it is obvious that, $\varepsilon(p)$ is negligible as claimed.

Lemma 1. Inverse of a combination function, i.e., $\frac{1}{(l\ n)}$ is negligible when $l >> n$.

Proof. We have, $\binom{l}{n} = \frac{l!}{n!(l-n)!}$ and $1/\binom{l}{n} = \frac{n!(l-n)!}{l!}$

$$= \frac{n!l!}{l!\prod_{i=0}^{n}(l-i)} = \frac{n!}{\prod_{i=0}^{n}(l-i)} = \frac{\prod_{i=1}^{n} i}{\prod_{i=0}^{n}(l-i)}$$

By plugging in $n = p$ and $l = p^2$ (ignoring the constant values), we get

$$= \frac{\prod_{i=1}^{p} i}{\prod_{i=0}^{p}(p^2-i)} = \frac{1}{p^2}\prod_{i=1}^{p}\frac{i}{(p^2-i)} \approx \frac{p!}{(p^2)^{p+1}} = \frac{p!}{p^{2p+2}}$$ because, for

large values of p we get, $p^2 >> p$. Thus, we have the function $f(n) = \frac{1}{p^{p+2}}$.

Now, applying the definition of the negligible function (Definition 2 above), we get $\forall c \in N$, $\lim_{p \longrightarrow \infty} \frac{1}{p^{p+2}} = 0$.

Proposition 1. If $f_1(n)$ and $f_2(n)$ are two negligible functions, then the function $f(n) = f_1(n) + f_2(n)$ is also negligible.

Based on the assumption that, retrieving a secret ***randomly embedded permutation*** is hard, specifically when the bit string is sufficiently large, the proposed scheme is CPA secure.

Analysis of Known Attacks:

The proposed scheme has been analyzed for the following known attacks:

1) *Brute-force attack*: Based on Theorem 1, it can be deduced that present scheme is secured against brute-force attack for all the cases where

$n << l$, l, since l is chosen as n^2. And since recovering the codeword actually requires the key K, then, $\forall K \subset [l]$, attack may be viewed as the brute-force against the secret key K. Guessing the key for sufficiently large parameters is considered a very difficult problem. Therefore, such an attack against the codeword would not be effective and can be thwarted with ease.

2) *Privacy of the homomorphic operations*: Privacy, in present scheme, is inherent due to a constant ciphertext size. This size does not change in course of operations over ciphertext. Hence it becomes hard to predict the type and number of operations performed.
3) *Attacks with respect to the first bit of codeword*: Attack over first bit is of no significance as the length of the codeword and locations of remaining codeword bits are not revealed. Deducing the first bit is if of no use in comparison to the total remaining length of codeword.
4) *Attacks against the DSCP*:Any such scheme can be defended by keeping the length of ciphertext$n^{5/2}$for a given codeword of$n$bits (18). Proposed scheme having the length of the ciphertext $l$which is larger than$n^{5/2}$, can defend itself against such attack.
5) *Attack based on the properties of the RM code*: It can be noticed that, generally the mapping between a message and a codeword in a RM code is deterministic. That means given a message m we always get the same codeword w. So, when a codeword is embedded in a random string using the same key$K$, we always get same bits at the positions specified by$K$ whenever we encrypt the same message m using the same key$K$. So, simple XOR operation over sufficient number of ciphertexts will reveal the positions at which the codeword is embedded, because, those positions contain zero bits after XORing. The only way this attack can be defended is that, the message to be encrypted should be changed in a way that encoding of the same message results in different codewords. Prepending a random string and zero bits as described in the *Encrypt* algorithm serves this purpose and with this message expansion operation such attacks can be successfully thwarted.
6) *Attack concerned with guessing ther, m values*: It is proposed to keep the Reed-Muller parametersr, m for additional security. But, it is possible to guess the r, mvalues as described below, though guessing them will not compromise the security of the scheme. As a simple analysis, consider the length of the ciphertext which is of$l$bits, where$l \geq n^2$. Unless $l >> n^2$, for smaller values we get$\sqrt{l} \approx n$or in other words the value of$\sqrt{l}$will be close to$n$. Since$n = 2^m$, it will be easy to guess the value of $n$as power of 2 and from that, it is easy to compute$m = log_2 n$. Now, having the value of m ther, mvalues can be any of the pairs$(1, m), (2, m), \ldots (m, m)$. So, one can consider these pairs in turn compute the corresponding generator matrix and obtain the plaintext corresponding to the codewords in each of these codes. However, the complexity of such attack to determine the codeword and in turn the corresponding plaintext is exponential, i.e.,2^nas discussed in Theorem 1 as probability. Moreover, the recovered plaintext each time would correspond to an expanded plaintext and obtaining the original plaintext from it adds up a further complexity of 2^k. Thus, the overall complexity will be2^{n+k}.

Conclusion:

In present work, □ a modified RMC-based symmetric key FHE scheme has been proposed with padding mechanism. The padding of random bit string to the plaintext has been ensuring one-to-many (non-deterministic) mapping between plaintext and ciphertext. It is shown that padding mechanism is enhanced the security against the IND-CPA. The modified algorithms of *Keygen*, *Encryption* and *Decryption* with respect to padding mechanism have been presented. The security of the proposed scheme is proved by Indistinguishability Chosen plaintext attack (IND-CPA) game-based proof. The mathematical proof of IND-CPA game-based security proof proved that retrieving a fixed length bit string which is embedded in the large random bit string specified by the secret positions is a hard problem. Further, the scheme is thoroughly analyzed with respect to the changes and new techniques suggested to show that the proposed scheme is secure against all the known attacks. The present scheme involves simple operations which makes this scheme easy to implement. The homomorphic operations *H.Add* and *H.Mul*are simple MOD 2 operations and they are easy to implement. In this paper, we focused on study of algorithm and its security proof with respect to the proposed enhancements. Many other aspects with respect to the practical implementation of the scheme with different parameters need to be studied and compared its performance analysis with other related works in future work.

Acknowledgement:

We thank the reviewers and editor of ICOCI 2021 for their helpful comments. This research received no specific grant from any funding agency in the public, commercial, or not-for profit sectors.

Authors' declaration:

- Conflicts of Interest: None.
- Ethical Clearance: The project was approved by the local ethical committee in Rajiv Gandhi University of Knowledge Technology.

References:

1. Yang L. Lihua L. Analysis of One Fully Homomorphic Encryption Scheme in Client-Server Computing Scenario, International Journal of Network Security. 2020;22(6),1032-1036.
2. Xu C. Cryptanalysis of an Improved Predicate Encryption Scheme from LWE. International Journal of Network Security. 2019;21(6)1054-1061.
3. Min ZE, YangG. Homomorphic encryption technology for cloud compu-ting. In: ICICT 2019 Procedia Computer Science. 2019;154,73-83.
4. Rivest R, Adleman L, & Dertouzos, ML. On data banks and privacy homomorphisms. In: Foundations of Secure Computation. 1978;169–180.
5. Rappe D. Homomorphic cryptosystems and their applications. Ph.D. thesis. University of Dortmund, Dortmund, Germany. 2004
6. Hallman R, Diallo M, August M, Graves C. Homomorphic Encryption for Secure Computation on Big Data, In: International Conference on Internet of Things, Big Data and Security (IoTBDS). 2018;340-347.
7. Anamaria V, Cosmin IN, Andrei P, Constantin S, Lucian MI. Applying Deep Neural Networks over Homomorphic Encrypted Medical Data. Computational and Mathematical Methods in Medicine. 2020. https://doi.org/10.1155/2020/3910250.
8. Challa R. Homomorphic Encryption: Review and Applications. In Advances in Data Science and Management. Springer. 2020;273-281.
9. Gentry C. Fully homomorphic encryption using ideal lattices. STOC ACM. 2009;169-178.
10. Gentry C. A fully Homomorphic Encryption scheme. [Ph.D Thesis]:Stanford University. 2009.
11. Brakerski Z, Vaikuntanathan V. Fully Homomorphic Encryption from Ring-LWE and Security for Key Dependent Messages. In: CRYPTO Springer Lecture Notes in Computer Science. 2011;505-524.
11. Dijk MV, Gentry C, Halevi S, Vaikuntanathan V. Fully Homomorphic encryption over the integers. In: Advances in Cryptology - EUROCRYPT'10 Springer Lecture Notes in Computer Science. 2010;6110,24–43.
12. Coron JS, Mandal A, Naccache D, Tibouchi M. Fully Homomorphic Encryption over the Integers with Shorter Public Keys. In: CRYPTO Springer Lecture Notes in Computer Science. 2011;6841,487-504.
13. Smart NP, Vercauteren F. Fully homomorphic encryption with relatively small key and ciphertext sizes. In: Public Key Cryptography - PKC'10 Springer Lecture Notes in Computer Science. 2010;6056,420–443.
14. Ramaiah YG, Gunta V. A New Fully Homomorphic Encryption Over The Integers Using Smaller Public Key. Int. J. Electronic Security and Digital Forensics Inder science. 2016;8(4),303–331.
15. Wang X, Tao L, Li J. A More Efficient Fully Homomorphic Encryption Scheme Based on GSW and DM Schemes. Security and Communication Networks. 2018.https://doi.org/10.1155/2018/8706940.
16. Amuthan A, Sendhil R. Hybrid GSW and DM based fully homomorphic encryption scheme for handling false data injection attacks under privacy preserving data aggregation in fog computing. J Ambient Intell Human Comput. 2020;11,5217–5231.
17. Challa R, Gunta V. Additively LWE based homomorphic encryption for compact devices with enhanced security. International Journal of Network Security. 2019;21(3),378–383.
18. Armknecht F, Augot D, Perret L, Sadeghi AR. On Constructing Homomorphic Encryption schemes from Coding Theory. In: IMACC Springer Lecture Notes in Computer Science. 2011;23-40.
19. Bogdanov A, Lee CH. Homomorphic encryption from codes. IACR Cryptology. 2011
20. Lee PJ, Brickell EF. An observation on the security of McEliece's public-key cryptosystem. In: Proceedings of EUROCRYPT'88.1988;275–280.
21. Applebaum B, Barak B, Wigderson A. Public-key cryptography from different assumptions. STOC. 2010;171–180.
22. Strenzke F. Message-aimed side channel and fault attacks against public key cryptosystems with Homomorphic properties. Cryptographic Engineering. 2011;1(4),283–292.
23. Berger TP, Cayrel PL, Gaborit P, Otmani. Reducing Key Length of the McEliece Cryptosystem. In: AFRICACRYPT Springer Lecture Notes in Computer Science. 2009;77-97.
24. Wang Y. Secure Random Linear Code Based Public Key Encryption Scheme RLCE. IACR Cryptology. 2015;298.
25. Loidreau P, Sendrier N. Weak keys in the McEliece public-key Cryptosystem. IEEE transactions of Information Theory. 2001;47(3),1207-1212.
26. Cheng-cheng Z, Ya-tao Y, Zi-chen L. The Homomorphic Properties of McEliece Public key Cryptosystem. In: Proceedings of IEEE ICMINS. 2012;39-42.
27. Challa R, Gunta V. Reed-Muller Code Based Symmetric Key Fully Homomorphic Encryption Scheme. In: Proceedings of 12th conference on Information Systems Security Springer Lecture Notes in Computer Science, 2016;10063,499-508.

28. Lee MS. Sparse subset sum problem from Gentry-Halevi's fully homomorphic encryption. IET Information Security. 2017;11(1),34-37.
29. Coron JS, Handschuh H, Joye M, Paillier P, Pointcheval D, Tymen C. GEM: □ Generic Chosen-Ciphertext Secure Encryption Method. In: Topics in Cryptology CT-RSA Lecture Notes in Computer Science Springer. 2002;2271, 263-276.
30. Mihaljevic MJ, Imai H. A Stream Cipher Design Based on Embedding of Random Bits. International Symposium on Information Theory and its Applications. 2008;1-6.
31. Le TP, Aono Y, Hayashi T, Wang L, Moriai S. Privacy-Preserving Deep Learning via Additively Homomorphic Encryption. IEEE Transactions on Information Forensics & Security. 2018;99,1–1.
32. Katz J, Lindell Y. Introduction to Modern Cryptography. CRC Press. 2007.

مفتاح متماثل معدَّل لمخطط تشفير متماثل الشكل تمامًا استنادًا إلى كود Read-Muller

راتناكوماري تشالا [1] **فيجيايا كوماري كونتا[2]**

[1]جامعة راجيف غاندي لتكنولوجيا المعرفة - أندرابراديش ، الهند
[2] جامعة جواهر لال نهرو التكنولوجية ، حيدر أباد ، تيلانجانا ، الهند.

الخلاصة:
أصبح التشفير المتجانس شائعًا وقويًا للتشفير لمختلف تطبيقات الحوسبة السحابية. حيث حدثت تطورات عديدة في العقود الأخيرة. تم اقتراح مخططات قليلة تستند إلى نظرية الترميز ولكن لا يدعم أي منها عمليات غير محدودة بأمان. نقترح مفتاحًا متماثلًا معدّلًا يعتمد على رمز Reed-Muller لتشفير متماثل تمامًا لتحسين أمانه باستخدام تقنية توسيع الرسائل. يوفر توسيع الرسالة باستخدام سلسلة ذات طول ثابت عشوائي مسبقة التعيين من واحد إلى متعدد بين الرسالة وكلمة التشفير ، وبالتالي تعيين واحد إلى العديد بين النص العادي والنص المشفر. يدعم المخطط المقترح عمليات الجمع والضرب (MOD 2) بشكل غير محدود. نحن نبذل جهدًا لإثبات أمان المخطط في ظل عدم القدرة على التمييز في ظل هجوم النص العادي المختار (IND-CPA) من خلال إثبات أمني قائم على اللعبة. يعطي دليل الأمان تحليلًا رياضيًا ومدى تعقيد الصعوبة. كما يقدم تحليلًا أمنيًا ضد جميع الهجمات المعروفة فيما يتعلق بتوسيع الرسالة والعمليات المتجانسة.

الكلمات المفتاحية: هجوم النص العادي المختار، تشفير متماثل الشكل، تبديل عشوائي، كود ريد مولر، مجموع مجموعة فرعية متفرقة.