computing@computingonline.net
www.computingonline.net

Print ISSN 1727-6209
On-line ISSN 2312-5381
International Journal of Computing

SECURE VIDEO TRANSCODING IN MOBILE CLOUD COMPUTING

Mohd Rizuan Baharon, Mohd Faizal Abdollah, Nur Azman Abu, Zaheera Zainal Abidin, Ariff Idris

Department of Computer System and Communication, Faculty of Communication and Information Technology, Universiti Teknikal Malaysia Melaka, Malaysia
{mohd.rizuan, faizalabdollah, nura, zaheera, miariff}@utem.edu.my

Paper history:
Received 17 July 2018
Received in revised form 18 September 2018
Accepted 2 October 2018
Available online 31 December 2018

Keywords:
Video Transcoding;
Fully Homomorphic Encryption;
Mobile Cloud Computing;
Data Security;
Data Privacy.

Abstract: Video Transcoding is one of the recent services available online nowadays provided by the clouds to enable a user to convert a video format from one into another in a very convenient way. To transcode a video, all of the video contents need to be uploaded to the cloud storage. However, outsourcing video contents that may contain sensitive information do not guarantee the video security and privacy as the clouds have the ability to access them. Thus, in this paper, an enhanced homomorphic encryption scheme is proposed to allow massive amount of frames to be transcoded by the cloud server in a secure manner. This scheme encrypts integers rather than individual bits so as to improve the scheme's efficiency. With the aid of a proposed process for multiple parties to communicate securely, the efficiency of the scheme is thoroughly evaluated and compared with related works. The result shows that our scheme offers much better efficiency, which makes it more suitable for operating the video transcoding in cloud environment.

1. INTRODUCTION

Cloud computing technology has emerged and attracted many private and public sectors including entertainment companies to transfer their in-house production to cloud servers [1]. The main reason for such movement is that more services such as video transcoding and 3D rendering are offered as pay as is used basis by the cloud providers [2, 3]. Video transcoding is a conversion process of a video format from one into another, along with a variety of solutions [2]. Video transcoding allows video content providers to offer different video formats to customers using various encoding techniques and resolutions. However, such a process requires huge processing resources and storage spaces to transcode and store the video data in multiple formats. Thus, by leveraging cloud facilities such as powerful computing resources and massive storage spaces, such a process could be executed efficiently without the need for spending more on upfront cost for the video transcoding facilities and storage servers.

Transcoding is essential to allow the video data to be stored and transmitted in a compressed format so that storage space and communication bandwidth can be reduced. To leverage the cloud facilities for video transcoding services, video content providers need to transfer the video content to the cloud servers. Many transcoding solution vendors such as Amazon Web Services (AWS), Zencoder and Panda provide real-time video transcoding services based on cloud computing. Their approaches seem to be reasonable, as video transcoding for a large number of clients needs a great amount of resources such as computing power, memory, and storage space [2, 4, 5].

Before transcoding takes place in the clouds, video data needs to be uploaded for storage. However, the confidentiality of the uploaded data becomes one of the primary concerns mainly if the data contains private information or video for commercial uses. Furthermore, revealing such video data to the untrusted third party like clouds may rise security concerns as the clients have very limited control over their data [6-8]. Moreover, improper managing such precious data may lead to a disaster to the data owner as a result of data misuse, data leakage, or data stolen by other parties that use the

same services [9, 10]. Based on the cloud nature together with the aforementioned reasons, addressing security and privacy issues in such an environment is a very challenging problem [11]. Thus, an encryption approach should be implemented to protect the outsourced data and to preserve the privacy of data owner in the clouds environment.

Existing encryption techniques such as AES encryption are good to protect the outsourced and stored data in the cloud storage [7]. Such an encryption could protect the confidentiality of the data without disclosing its content to the unauthorised users. Nevertheless, such a technique prevents the data from being processed by the cloud processor. As a result, it is almost impossible to adapt them to special video application paradigms which pose special requirements that are never encountered when encrypting text data [12]. Thus, a new encryption scheme needs to be proposed to allow encrypted data to be processed without decryption.

An encryption scheme that allows data to be processed in an encrypted form has been introduced in [13] and is known as a Fully Homomorphic Encryption (FHE) scheme. Since then, a lot of FHE schemes have been proposed and improved upon due to the scheme's efficiency issue for their implementation [14, 15]. Such a limitation requires an improved FHE scheme to be proposed. To address the above problem, in this paper, we propose a new lightweight homomorphic encryption scheme suitable for transcoding video contents in a secure manner. This scheme encrypts integer and produces encrypted result in the integer form. Such a setting improves the transcoding process as encryption on an integer is faster than encryption on every single bit of the integer [16]. Moreover, encryption over the integer increases input and output message spaces so as to consume less storage space and requires less bandwidth for data transmission.

To the best of our knowledge, we are the first to apply a fully homomorphic encryption scheme to allow video transcoding to be processed in the cloud environment. Thus, the main goal of this paper is to propose a new lightweight fully homomorphic encryption scheme for mobile cloud-based video transcoding. This goal can be achieved by utilising a symmetric encryption scheme that uses a secret key to achieve the balance of efficiency and security [17]. Such a balance of efficiency can be demonstrated by investigating the delay of the whole process of video transcoding on encrypted frames while the scheme's security can be illustrated by providing the security analysis on the proposed scheme. Investigation on the delay of such a process can be executed by using a network simulation software such as OPNET. Furthermore, this scheme is designed to enable the MPEG compression technique to be processed with the encrypted Discrete Cosine Transform coefficients. Such a technique is widely used in multimedia data for entertainment as well as business purposes [5].

The rest of this paper is briefly described as follows. Section 2 explains the background of the related works. Section 3 gives the details about MPEG compression technique to compress a video into MPEG format. Section 4 describes the details of the proposed cryptosystem. Section 5 explains the security analysis of the proposed cryptosystem. Section 6 explains the details of video transcoding in the cloud environment. In Section 7, the application settings and experimental results of our scheme are given in detail, and discussions on the scheme performance are presented. Finally, the conclusion is given in section 8.

2. RELATED WORKS

2.1 SECURITY OF MULTIMEDIA DATA AND CLOUD COMPUTING

B. Saeed, and N. Majid, have proposed the use of simple and lightweight stream cipher algorithm to secure the multimedia data after taking into consideration the fact that such data contains excess volume of information and needs real-time uses [5]. To secure the data by means of encryption, additional computation is needed. Thus, the security and the necessity have to be balanced. This encryption was proved to be secured by C. E. Shannon in 1949, but the key stream must be generated completely at random with at least the same length as the plaintext and cannot be used more than once. Such requirement makes the scheme very trivial to be implemented in practice, and as a result, the schemes have not been widely used except for the most critical applications [18].

Furthermore, according to the survey made by L. Fuwen, and K. Hartmut, many encryption algorithms proposed operating after compression. Only two of them operated before the compression schemes, which were the Pazarci-Dipcin scheme and the correlation-preserving video encryption scheme. However, both of them were proven unsecure enough due to the former was not secure against brute force attacks, and the known or chosen-plaintext attacks, while the latter was not secure against known-plaintext attacks. Moreover, the latter scheme has great limitation as it is merely applied to video codecs that use only intra-frame technology, such as M-JPEG. It cannot be deployed for the

widely used video codecs that apply the hybrid coding technologies, such as MPEG-2 and H.264 [12].

In other research works, several approaches have been proposed to avoid decryption of protected multimedia content at mid-network nodes. Mou et al., have designed a secure media streaming mechanism by making use of the existing highly studied cryptographic techniques. A secure media streaming mechanism has been proposed, which combines encryption, authentication, and transcoding to address content protection, sender authentication, and media adaptation, respectively, and coherently. However, their scheme cannot be implemented in a cloud computing environment as they assumed mid-network proxies are trusted devices, so decryption can also be done on mid-network proxies for the purpose of transcoding. This contradicts with our assumption as the cloud service provider is an untrusted party. The cloud providers are only responsible for transcoding job and they are supposed not to see the content of the video they are processing [19].

2.2 FULLY HOMOMORPHIC ENCRYPTION SCHEME AND ITS EFFICIENCY

Most of the existing FHE schemes are suffering from an efficiency issue as their choice of plaintext for encryption and the generated ciphertext is in the form of bits [13, 20, 21]. The advantage of such individual bit based encryption is easier to achieve due to fully homomorphic properties. Nevertheless, these encryption schemes significantly reduce the storage and communication efficiency that leads to an increase in the computational time. Furthermore, the schemes require applications to convert computation tasks into binary addition and multiplication operations, which makes the computation more complex [16]. Furthermore, several schemes have been proposed so that the plaintext for encryption is in the form of integers, while the output remains in the form of bits [22-24]. Those schemes support arbitrary functions in an encrypted form with better efficiency as they are designed on the basis of integers. Nevertheless, such schemes are also hardly to be implemented by resource-constraint devices due to still high computational complexity and communication costs [16]. In addition, those schemes require a large public key size for encryption [20, 25], which rapidly reduces the battery lifetime of mobile devices during data encryption.

In the recent work of H. Zhou and G. Wornell [16], a new homomorphic encryption scheme has been developed. The scheme operates directly on integer vectors that support three operations, which are more specifically implemented in signal processing applications. The operations supported by this scheme are addition, linear transformation and weighted inner products. However, such a scheme has a limitation on the degree of a polynomial to be computed efficiently. Furthermore, the scheme suffers from an efficiency issue due to the adopted large public key size [16]. In addition, in the work [25] a scheme with both plaintext and ciphertext in the form of integers has been proposed. This approach improves the scheme's efficiency as it has been discussed earlier. In our view, this is the only scheme that considers both plaintext and the generated ciphertext in the integer form. It allows arbitrary functions to be executed on encrypted data. Nevertheless, the scheme is designed for devices with higher performance power due to both plaintext and ciphertext data being represented as matrices. Processing and transmitting data in a matrix form requires more computing resources and bandwidths as well as more storage.

3. MPEG VIDEO COMPRESSION TECHNIQUE

Raw video contains a large amount of data. On the other hand, communication and storage capabilities are limited and thus expensive. For example, a given HD video signal might have $720 * 1280\ pixels/frame$, and a playback speed of 40 frames/sec and this produces an information flow of:

$$\frac{720*1280 pixels}{frame} \times \frac{40\ frames}{sec} \times \frac{3\ colors}{pixel} \times \frac{8\ bits}{color} = 884.74 Mb/s. \quad (1)$$

For a channel with bandwidth $50 Mb/sec$, it requires the video to be compressed by a factor of about 18. The way this is achieved is through video compression. Video compression is done through reduction of redundancy and irrelevancy.

3.1 THE OVERVIEW

The MPEG bit-stream structure can be showed in an abstract way as in Fig. 1. The Figure shows the bit-stream structure that results from video compression algorithms. The 8x8 block values are coded by means of discrete cosine transform.

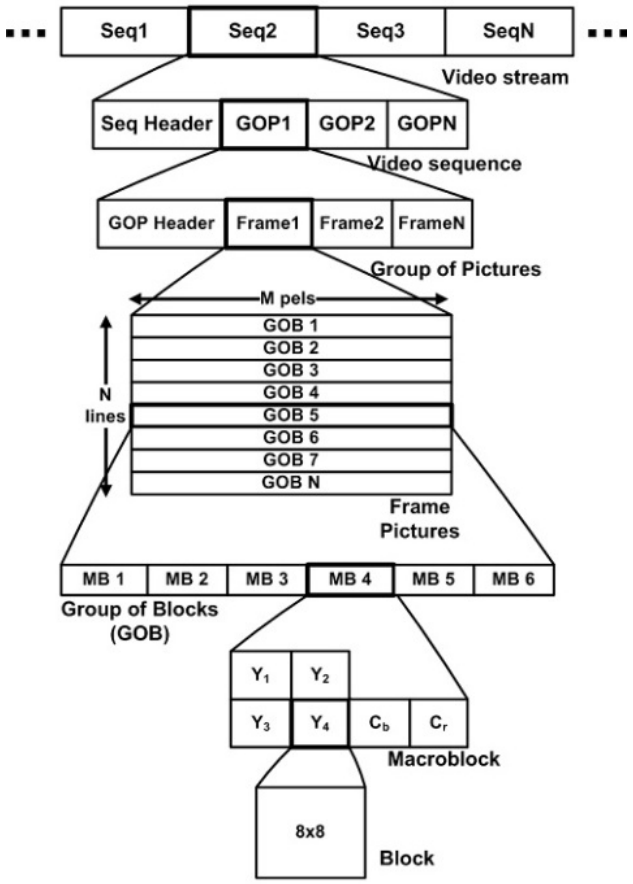

Figure 1 – MPEG Codecs video in a hierarchy of layers

3.2 THE DCT AND IDCT

The Discrete Cosine Transformation (DCT) Formula is a technique to compress MPEG Video format. The DCT is one of the most popular transforms used in multimedia compression. According to Equation 2 in two-dimensional condition, the DCT operates on N by N block of pixels $f(x,y)$, and its output is blocks with N by N block of pixels

$$F(u,v) = \frac{2}{\mathrm{N}} C_u C_v \sum_{y=0}^{N-1} \sum_{x=0}^{N-1} f(x,y) \cos\left[\frac{(2x+1)u\pi}{16}\right] \cos\left[\frac{(2y+1)v\pi}{16}\right], \quad (2)$$

where

$$C_u = \begin{cases} \frac{1}{\sqrt{2}} & \text{if } u = 0, \\ 1 & \text{if } u > 0 \end{cases} \quad ; C_v = \begin{cases} \frac{1}{\sqrt{2}} & \text{if } v = 0, \\ 1 & \text{if } v > 0 \end{cases}.$$

From Equation 2, $f(x,y)$ is the brightness of the pixel at position $[x,y]$. $F(u,v)$ is a set of N by N coefficients representing the data in the transformed matrix value at position $[u,v]$. A set of waveforms is defined for each possible value of N (usually $N = 8$, thus, there exists 64 waveforms). Each coefficient can be seen as the weight of each of these basis patterns or waveforms. By summing all the waveforms scaled by the corresponding weight, the original data can be recovered [18].

The Inverse Discrete Cosine Transform IDCT formula is given below:

$$f(x,y) = \sum_{u=0}^{7} \sum_{v=0}^{7} F(u,v) \frac{C_u}{2} \frac{C_v}{2} cos\left[\frac{(2x+1)u\pi}{16}\right] cos\left[\frac{(2y+1)v\pi}{16}\right], \quad (3)$$

where $F(u,v)$ is a transform matrix value at position $[u,v]$ and $f(x,y)$ is the original pixel of the video content as described above. The IDCT is used by the decoder to reconstruct the pixel values of the compressed video.

4. THE PROPOSED SCHEME

The proposed scheme employs its key generation algorithm for Data User (DU) to receive a unique symmetric secret key from Video Contributor (VC). The data encryption algorithm of the scheme allows VC to encrypt its contributed video with the key and outsources the encrypted video to CS for transcoding process. This enables all the VC to contribute their video to CS in an encrypted form, which CS is unable to decrypt. The video transcoding and recovery algorithm of the scheme allows CS to transcode the received video without decryption based on a desired format requested by DU and transmit the result to DU. DU then decrypts the received result to recover its plaintext video. The details of these three algorithms are given below.

4.1 KEY GENERATION

The proposed scheme employs a secret key for data encryption by VC. The secret key is shared only between its associated DU and VC, and used for symmetric data encryption.

To produce this key, the parameter delineations for the verifiable encryption of RSA signatures are adopted [26]. That is, DU defines φ as the product of two safe primes p and w, i.e., $\varphi = pw$ where $p = 2p' + 1$ and $w = 2w' + 1$ with p' and w' being primes. φ will be used as a public number, w needs to be discarded without disclosing it to anyone, and p should be kept securely. Additionally, DU selects a prime r ($< p$) and stores both p and r as its secret master keys.

To generate a key for VC, DU picks up random numbers $s_i < p$ and $q_i > w$ to produce the following symmetric secret key:

$$k_i = (rs_i + pq_i) \bmod \varphi, \quad (4)$$

and k_i is only given to VC as its secret key. The n secret keys k_i need to meet the following conditions:

1. For summation,
   $2^{l_d} n < r$ and $r\left(1 + 2^{l_{\tilde{c}}+l_s} n\right) < p$.
2. For multiplication,
   $2^{nl_d} < r$ and $\left(2^{l_d} + 2^{l_{\tilde{c}}+l_s} r\right)^n < p$.

Here, l_d, $l_{\tilde{c}}$ and l_s are the maximal bit lengths of data item d_i, a random number $\tilde{c}_j$ chosen by VC for its data encryption, and random number s_i in key k_i for any i, respectively. The detailed reasons for the above conditions will be discussed later when the proposed data encryption and decryption are presented. In brief, the first part of both conditions says that the sum or the product of encrypted data items is less than r for the purposes of ensuring the recovery of the sum or product result. The second part of both conditions means that the calculation on the first part of each u_i's secret key together with the other items results in a number less than p. This condition allows the summation or product result to be recovered.

4.2 DATA ENCRYPTION

In this sub-section, an algorithm is presented to allow VC to generate its encrypted data for submission to CS. To do so, VC first performs the following calculation:

$$\tilde{\alpha}_i = \left(d_i + \tilde{c}_j k_i\right) \bmod \varphi. \quad (5)$$

Here, $\tilde{\alpha}_i$ is the encrypted form of video pixel data item d_i, and $\tilde{c}_j$ is a random number picked up by VC for each data item d_i. After the completion of the above calculation, VC sends $\tilde{\alpha}_i$ to CS for storing and computing purposes. Upon the receipt of completed $\tilde{\alpha}_i$ from VC, CS starts its computation on the ciphertexts received to generate a specific video format requested by DU.

4.3 VIDEO TRANSCODING AND RECOVERY

In this sub-section, algorithms are specified for CS to compute the received video data using the required operations without decryption. Once the transcoding process is completed, the compressed video will be sent to DU for decryption. The following are the steps for adding and multiplying of ciphertext data. In addition, how the algorithms support homomorphism under both addition and multiplication is also explained. Homomorphism under these operations is defined below:

Definition 4.1:

Let $*_G$ and $*_H$ be arbitrary operations in groups G and H, respectively. A function $f: G \to H$ from group G to group H is a (group) homomorphism if the group operation is preserved in the sense that:

$$f(g_1 *_G g_2) = f(g_1) *_H f(g_2). \quad (6)$$

For all $g_1, g_2 \in G$, let e_G be the identity in G and e_H the identity in H. A group homomorphism f maps e_G to e_H: $f(e_G) = f(e_H)$. Note that f must preserve the inverse map due to:

$$f(g)f(g^{-1}) = f(gg^{-1}) = f(e_G),$$

therefore: $f(g^{-1}) = f(g)^{-1}$.

i. *Summation*

Let $f(d_1, d_2, \ldots d_n) = \sum_{i=1}^{n} d_i$, i.e. the summation of all the data items d_i for $1 \le i \le n$. For summing n ciphertext in MCC, CS computes the received ciphertext data for each frame as follows:

$$f(\tilde{\alpha}_1, \tilde{\alpha}_2, \ldots, \tilde{\alpha}_n) = \left(\sum_{i=1}^{n} \tilde{\alpha}_i\right) \bmod \varphi.$$

Then, this result will be sent to DU for recovering the sum. To obtain the sum, DU applies its master keys p and r to calculate:

$$\sum_{i=1}^{n} d_i = \left(f(\tilde{\alpha}_1, \tilde{\alpha}_2, \ldots, \tilde{\alpha}_n) \bmod p\right) \bmod r. \quad (7)$$

ii. *Product*

Let $\tilde{\alpha}_i$ and $\tilde{\alpha}_j$ be the ciphertexts of plaintexts d_i and d_j, respectively. The product of d_i and d_j can be recovered from the product of $\tilde{\alpha}_i$ and $\tilde{\alpha}_j$ as follows:

$$d_i d_j = \left(\left(\left(\tilde{\alpha}_i \tilde{\alpha}_j\right) \bmod \varphi\right) \bmod p\right) \bmod r. \quad (8)$$

Based on the FHE concept given in Definition 4.1, it is clear that our scheme (or its algorithms) is homomorphic under both addition and multiplication operations.

5. SECURITY ANALYSIS

In this section, the security of the scheme proposed in the previous sub-section is analysed. We analyse the scheme security based on a brute force attack on Video Contributor's (VC) secret keys.

The proposed scheme is said to be secured against a brute force attack on a VC's secret key due to the following reason: given $\tilde{\alpha}_i$ and φ, an attacker with some knowledge about the plaintext related to $\tilde{\alpha}_i$ is unable to retrieve any useful information for successfully inferring the encryption key. Such a

claim can be achieved by adding a random parameter $\tilde{c}_j$ in ciphertext $\tilde{\alpha}_i$. Such a parameter can improve the security of the encrypted data by avoiding any information about the encryption key being disclosed to unauthorised users.

Such an attack can be elaborated as follows. Suppose that the encryption algorithm for plaintext d_i with secret key k_i is:

$$\tilde{\alpha}'_i = (d_i + k_i) \bmod \varphi.$$

Such an encryption algorithm is vulnerable against a brute-force attack on key k_i. The reason is that, when the size of d_i is small compared to key k_i used for the encryption, the high-end part of the ciphertext generated is likely to be identical to that of the key. This means that for different encryptions with the same key, the differences among them are just the bits at the lower end of the ciphertexts, while the rest (the higher end of the key) remains the same. In case the attacker is able to obtain several ciphertexts, it can compare them to spot their identical part so as to gain that part of the key. If the remaining part of the key is short, then the attacker can guess it by a brute force attack.

Thus, to prevent such an attack, a unique random parameter $\tilde{c}_j$ is added in Equation 2 for our encryption. By adding this parameter, attempting to spot any identical part of the key will be intractable as it will be hidden by $\tilde{c}_j$, which is known only by VC.

6. SECURE VIDEO TRANSCODING IN THE CLOUD

6.1 COMPUTATION ON ENCRYPTED FLOATING POINT NUMBER

Most of the existing cryptosystems are incompatible with floating point numbers mainly when the cryptosystem uses modulo operation on the integer. This is due to the fact that modulo operation on the integer will always return the output as an integer form. In our proposed cryptosystem, we offer a cryptosystem that can compute floating point numbers by using an appropriate approach as described below.

Multiplication of a floating point number by encrypted data

Suppose an integer d is encrypted using the proposed scheme, $E_p(d)$. To multiply the ciphertext $E_p(d)$ by a fraction $\frac{1}{b}$, the steps below need to be followed:

- Determine the precision, d. For instance, let us consider the precision, d is two.
- Represent $\frac{1}{b}$ as a floating point number with d – precision, $y = 0.xx$.
- Multiply $y = 0.xx$ by 10^2 to change it into integer form $\bar{y} = xx \in \mathbb{Z}$.
- Multiply the ciphertext by the integer $\bar{y} = xx$, $(E_p(d) \times \bar{y})$. All operations are done in modulo n for security reason.
- Decrypt the result using the scheme decryption algorithm, $\left[\left(E_p(d) \times \bar{y}\right)\right] mod\ p$.
- Multiply the result in the plaintext form by 10^{-2}:

$$(d \times \bar{y}) \times 10^{-2} = d \times \frac{1}{b}. \qquad (9)$$

Multiplication of two ciphertexts which both of the original data are floating point numbers

Let d_1, and d_2 be two fractions and $E_p(d_1)$, and $E_p(d_2)$ be the ciphertexts of d_1, and d_2 respectively. These two numbers can be multiplied in an encrypted form accordingly to the following steps.

- Determine the precision of d_1, and d_2. Let us consider that the precision of both plaintext is 2. Then both plaintexts are multiplied by 10^2 to change them into the integer form $\overline{d_1}$, and $\overline{d_2}$.
- Encrypt $\overline{d_1}$, and $\overline{d_2}$ and represent them as vectors $\{(E_p(\overline{d_1}), 10^2), (E_p(\overline{d_2}), 10^2)\}$.
- Multiply homomorphically these two vectors as follows:

$$\left(E_p(\overline{d_1}), 10^2\right) \times \left(E_p(\overline{d_2}), 10^2\right) = \left(E_p(\overline{d_1} \times \overline{d_2}), 10^4\right).$$

- Decrypt the encrypted result using the decryption algorithm:

$$\overline{d_1} \times \overline{d_2} = \left(E_p(\overline{d_1} \times \overline{d_2})\right) mod\ p\ .$$

- Compute the result of $d_1 \times d_2$ by multiplying $\overline{d_1} \times \overline{d_2}$ with 10^{-4}:

$$\left(\overline{d_1} \times \overline{d_2}\right) \times 10^{-4} = d_1 \times d_2.$$

Combination of multiplication and addition on ciphertexts which all the original data are floating point numbers

Let d_1, d_2 and d_3 be three fractions and $E_p(d_1)$, $E_p(d_2)$ and $E_p(d_3)$ be the ciphertexts of d_1, d_2 and d_3 respectively. To multiply two ciphertexts and add

another ciphertext, the steps below need to be followed:

- Determine the precision of those plaintexts, d. Let us consider that the precision of the plaintexts is 2. Then for multiplication purposes, the first two plaintexts need to be converted into integers $\overline{d_1}$, and $\overline{d_2}$ by multiplying both of them by 10^2.
- Encrypt the plaintexts and represent them as vectors $\{(E_p(\overline{d_1}), 10^2), (E_p(\overline{d_2}), 10^2), (E_p(\overline{d_3}), 10^2)\}$.
- Multiply homomorphically the first two vectors as follows:
$$(E_p(\overline{d_1}), 10^2) \times (E_p(\overline{d_2}), 10^2) = (E_p(\overline{d_1} \times \overline{d_2}), 10^4).$$
- Add the multiplication result to $(E_p(\overline{d_3}), 10^2)$. Prior addition can take place, the tenth of both vectors must be the same. Thus, $(E_p(\overline{d_3}), 10^2)$ have to be multiplied by 10^2. $\{(E_p(\overline{d_1} \times \overline{d_2}), 10^4), (E_p(\overline{d_3}) \times 10^2, 10^4)\}$.
- Add these vectors homomorphically as follows:

$$(E_p(\overline{d_1} \times \overline{d_2}), 10^4) + (E_p(\overline{d_3}) \times 10^2, 10^4) = \left(E_p\left((\overline{d_1} \times \overline{d_2}) + \overline{d_3} \times 10^2\right), 10^4\right).$$

- Decrypt the result using the scheme decryption algorithm:

$$(\overline{d_1} \times \overline{d_2}) + \overline{d_3} \times 10^2 = \left(E_p\left((\overline{d_1} \times \overline{d_2}) + \overline{d_3} \times 10^2\right)\right) mod\ p\ .$$

- Compute $(d_1 \times d_2) + d_3$ by multiplying $(\overline{d_1} \times \overline{d_2}) + \overline{d_3} \times 10^2$ with 10^{-4}:

$$\left((\overline{d_1} \times \overline{d_2}) + \overline{d_3} \times 10^2\ \right) \times 10^{-4} = (d_1 \times d_2) + d_3). \quad (10)$$

6.2 SECURE MPEG VIDEO COMPRESSION TECHNIQUE

Such approaches that have been described in section 6.1 allow the MPEG video compression technique to be processed by the clouds securely without revealing any content of the video to the cloud providers. To illustrate how the process is done in the ciphertext form, let us consider the following computation on a block of frame of the size of 8×8 pixels that has widely used in MPEG compression technique. In this case, let us consider the following 8×8 bar diagram and black-white frame as shown in Fig. 2:

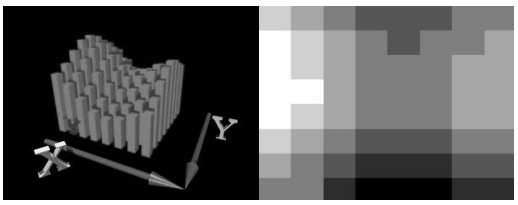

Figure 2 – Bar diagram and Black-white frame

$$A = \begin{pmatrix} a_{0,0} & a_{0,1} & \dots & a_{0,7} \\ a_{1,0} & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ a_{7,0} & \dots & \dots & a_{7,7} \end{pmatrix}.$$

The bar diagram and black-white frame as shown in Fig. 3 are normally used to represent the brightness of each pixel in 8×8 block. Both of the diagrams can be represented as a matrix block A. In order to compress the matrix block, the DCT formula will be implemented. To calculate the first element in the compressed block, we implement the formula as shown in equation 2. Since our $N = 8$, and to find the first value of the compressed data at $u = 0, v = 0$, equation (2) can be expressed as

$$F(0,0) = \frac{1}{4}\sum_{j=0}^{7}\sum_{x=0}^{7} f(x,y). \quad (11)$$

As the formula involved a fraction $\frac{1}{4}$, we have to restructure the equation using our approach as described above by converting the fraction into integer as follows:

$$F(0,0) = \frac{0.25\times10^{s}\times\left(\sum_{y=0}^{7}\sum_{x=0}^{7} f(x,y)\right)}{10^{s}}, \quad (12),$$

where $s = 2$.

In order to compress the block securely, each element $f(x,y)$ needs to be encrypted $E_{p,r}(f(x,y))$. All the encrypted data will be partially computed as follows:

$$\overline{E(F(0,0))} = 5 \times \left(\sum_{y=0}^{7}\sum_{x=0}^{7} E_{p,r}(f(x,y))\right). \quad (13)$$

The result of this computation $\overline{E(F(0,0))}$ needs to be decrypted before it is divided by 100:

$$E_{(p,r)}{}^{-1}\left(\overline{E(F(0,0))}\right) = F(0,0) \times 100. \quad (14)$$

In order to get the first element in plaintext form, we have to divide the result by 100:

$$F(0,0) = \frac{(F(0,0)\times 100)}{100}. \quad (15)$$

For other coefficient in the block, the similar way is applied to compress it using our proposed cryptosystem.

7. PERFORMANCE EVALUATION

The performance of our proposed scheme is evaluated in this section based on our experimental results and analysis. Application or experimental settings required by two distinct simulation software packages Matlab version 15a and OPNET version 14.5.A will be described first. The purpose of Matlab is for data computation, while OPNET is for simulating data transmission. Then, our experimental results and their analysis are presented, which provide meaningful evidence to support the conclusions provided at the end of this section.

7.1. APPLICATION SETTINGS

Cloud computing allows data to be outsourced to reduce the burden of data processing internally. Nevertheless, the security and privacy of the outsourced data cannot be scarified and need to be protected [27-31]. Furthermore, the efficiency of the implemented encryption scheme needs to be considered as transcoding process requiring excessive computation on the video data before the outsourcing demanding a huge processing resources of video contributor [32]. Thus, to have a balance scheme in term of security and computation complexity, this sub-section describes the application settings for implementing the proposed scheme.

To process video data in its ciphertext form, the workflow of MCC is illustrated in Fig. 3.

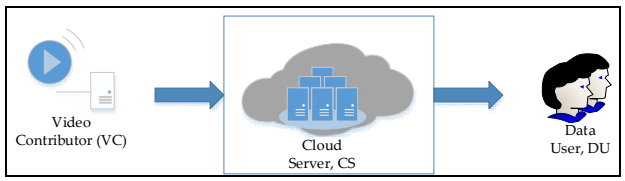

Figure 3 – The Proposed MCC Workflow

As set out in Section 3, there are *VC*, *DU* and *CS* managed by its CSP with their responsibilities elaborated below:

1. *VC*: It is a Video Contributor company, which provides a video that requires to be transcoded in multiple formats as requested by its clients. With low computing resources and storage spaces, such a company needs to leverage the transcoding facilities provided by the cloud.
2. *DU*: They are a group of mobile devices like smart phones or tablets. All the devices are connected to the Internet through wireless connections. The group members receive various formats of video prepared by CS after necessary transcoding process.
3. *CS*: It is a cloud server managed by a CSP, which provides transcoding services to its clients. It is an untrusted party to provide Internet based applications and deliver services through Internet connections. *VC* takes advantage of such services to ask CS to transcode a video to a various video formats requested by *DC*.

7.2. PERFORMANCE EVALUATION

For performance evaluation, we have selected a scheme as in [23] to be compared with the proposed scheme. This scheme that we named as a Fully Homomorphic Encryption over Matrix (FHEM) form has better efficiency for its implementation as this scheme encrypts integer. The details of the FHEM scheme can be found in [25]. The comparison results demonstrate the merit of our lightweight scheme in terms of its efficiency.

7.2.1 EXPERIMENTAL SETUP

In this sub-section, the delay of one round video transcoding process is measured and compared. The two schemes with various numbers of frames have been implemented. The parameter settings of the schemes are given in the next sub-section, while the results and discussions of the conducted experiments are given in Sub-section 7.2.3.

7.2.2 PARAMETERS SETTINGS

In our experiments, we use the parameter settings shown in Table 1.

Table 1. The Parameter Settings (Length in Bits)

Parameters	Our Scheme	FHEM Scheme [25]
p	664	1024
w	360	N/A
r	80	16
l_d	10	10
$l_{\tilde{c}}$	80	N/A
l_s	80	N/A
q_i	1024	N/A
φ	1024	N/A

7.2.3 RESULTS AND DISCUSSION

For experimental purposes, the results of the tests with respect to the various numbers of frames to be transcoded are illustrated in Table 2.

Table 2: Total Execution Time for frame processing

Number of Frames	Delay in Hours		Percent Difference $(= \frac{\lvert x_i - y_i \rvert}{\frac{(x_i + y_i)}{2}} \times 100\%)$
	Our Scheme, x_i	FHEM, y_i	
20000	0.21	22.23	196.25
40000	0.42	44.46	196.25
60000	0.64	66.69	196.22
80000	0.85	88.92	196.22
100000	1.06	111.15	196.22
120000	1.27	133.38	196.22
140000	1.48	155.61	196.22
160000	1.70	177.84	196.21

Fig. 4 demonstrates the delay of one round video transcoding process on encrypted frames by the two schemes.

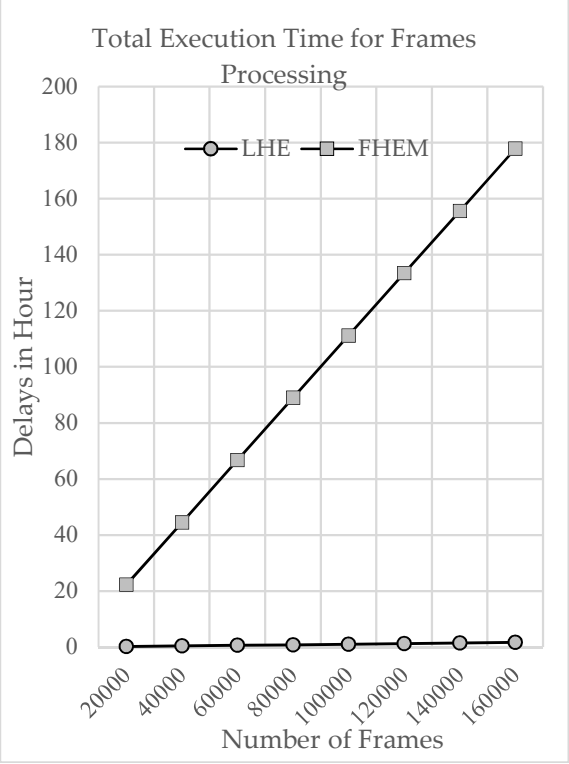

Figure 4 – Total execution time for one round of video data processing in the ciphertext form

The difference from the two lines shows that the delay introduced by our scheme gradually increases as the number of transcoded frames increases. However, for FHEM, its delay is over 50 times higher than our scheme and goes higher as the number of frames gets larger. For example, when the number of frames is 50000, the delay of the transcoding process by using our proposed scheme is below 0.1 hour, whereas the delay by FHEM is nearly 50 hours. Furthermore, when the number of frames increases to 400000, the delay caused by our scheme is still lower, which is around 4 hours, whereas FHEM takes longer than 400 hours. The main reason for the above differences is that FHEM involves the matrix multiplication of keys and data [25]. Such computation incurs cubic complexity on each encrypted frame prior to frames processing and hence extra delays.

8. CONCLUSION

In this paper, we have proposed a new lightweight homomorphic encryption scheme to allow video transcoding to be leveraged securely and efficiently in the cloud environment. The security analysis has confirmed the security strength of the scheme. Further analysis of the proposed scheme has shown that it does not only secure the transcoding process in the cloud environment, but it has also improved the performance of the process as our scheme operates much faster than the most relevant existing works, thanks to its lower complexity in terms of computations. In addition, to allow the video data to be processed in an encrypted form without having any difficulties, we have proposed an approach for our scheme to deal with floating point numbers. This is essential as DCT formula consists of computation on floating point numbers. To the best of our knowledge, we are the first to offer an encryption scheme that enables encryption of floating point numbers, computation on encrypted floating point numbers and return the answer in the floating point numbers. Moreover, the scheme achieves good simplicity and high efficiency as it is designed on the basis of integers. These merits allow transcoding services provided by CSPs can be executed in a more efficient and secure manner.

9. ACKNOWLEDGEMENT

The authors would like to thank to Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka (UTeM), Centre for Advanced Computing Technology, C-ACT and INSFORNET research group for their incredible supports in this project.

REFERENCES

[1] V. Turchenko, V. Shults, I. Turchenko, R. M. Wallace, M. Sheikhalishahi, J. L. Vazquez-Poletti, and L. Grandinetti, "Spot Price prediction for cloud computing," *International Journal of Computing*, vol. 12, issue 4, pp. 348–359, 2013.

[2] F. Jokhio, A. Ashraf, S. Lafond, and J. Lilius, "A computation and storage trade-off strategy for cost-efficient video transcoding in the cloud," *Proceeedings of the 39th Euromicro Conference on Software Engineering and Advanced Applications*, Sep. 2013, pp. 365-372.

[3] Z. Kuang, S. Guo, J. Liu, and Y. Yang, "A quick-response framework for multi-user computation offloading in mobile cloud

computing," *Future Generation Computer Systems*, vol. 81, pp. 166–176, 2018.

[4] S. Ko, S. Park, and H. Han, "Design analysis for real-time video transcoding on cloud systems," *Proceedings of the 28th Annual ACM Symposium on Applied Computing - SAC'13*, 2013, p. 1610.

[5] A. Ashraf, F. Jokhio, T. Deneke, S. Lafond, I. Porres, and J. Lilius, "Stream-based admission control and scheduling for video transcoding in cloud computing," in *Proceedings of the 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*, 2013, pp. 482-489.

[6] M. Sookhak, F. R. Yu, M. Khurram, and Y. Xiang, "Attribute-based data access control in mobile cloud computing : Taxonomy and open issues," *Future Generation Computer Systems*, vol. 72, pp. 273–287, 2017.

[7] M. R. Baharon, Q. Shi, and D. Llewellyn-Jones, "A new lightweight homomorphic encryption scheme for mobile cloud computing," *15th IEEE International Conference on Computer and Information Technology, CIT 2015, 14th IEEE International Conference on Ubiquitous Computing and Communications, IUCC 2015, 13th IEEE International Conference on Dependable, Autonomic and Secure*, 2015.

[8] J. Qi, M. Jienfeng, and W. Fushan, "On the Security of a Privacy-Aware Authentication Scheme for Distributed Mobile Cloud Computing Services," *IEEE System Journal*, vol. PP, no. 99, pp. 1–4, 2016.

[9] L. Griebel, H. Prokosch, F. Köpcke, D. Toddenroth, J. Christoph, I. Leb, I. Engel, and M. Sedlmayr, "A scoping review of cloud computing in healthcare," *BMC Medical Informatics and Decision Making*, vol. 15, no. 17, pp. 1–16, 2015.

[10] O. Zibouh, A. Dalli, and H. Drissi, "Cloud Computing security through parallelizing fully homomorphic encryption applied to multi-cloud approach," *Journal of Theoretical and Applied Information Technology*, vol. 87, no. 2, pp. 300–307, 2016.

[11] M. R. Baharon, Q. Shi, D. Llewellyn-Jones, and M. Merabti, "Secure rendering process in cloud computing," *Proceedings of the 11th Annual Conference on Privacy, Security and Trust, PST 2013*, 2013, pp. 82–87.

[12] F. Liu and H. Koenig, "A survey of video encryption algorithms," *Computers & Security*, vol. 29, no. 1, pp. 3–15, Feb. 2010.

[13] C. Gentry, *A Fully Homomorphic Encryption Scheme*, Ph.D. Dissertation, Stanford University, 2009.

[14] M. Naehrig, K. Lauter, and V. Vaikuntanathan, "Can homomorphic encryption be practical?," *Proceedings of the 3rd ACM workshop on Cloud computing security workshop CCSW'11*, 2011, p. 113.

[15] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A survey on homomorphic encryption schemes: theory and implementation," *Journal ACM Computing Surveys (CSUR)*, vol. 50, no. 6, pp. 83:2-83:33, 2017.

[16] H. Zhou and G. Wornell, "Efficient homomorphic encryption on integer vectors and its applications," in *Information Theory and Applications Workshop (ITA)*, 2014, pp. 1–9.

[17] R. Masram, V. Shahare, J. Abraham, and R. Moona, "Analysis and Comparison of symmetric key cryptographic algorithms based on various file features," *International Journal of Network Security & Its Applications (IJNSA)*, vol. 6, no. 4, pp. 43–52, 2014.

[18] S. Bahrami and M. Naderi, "Encryption of multimedia content in partial encryption scheme of DCT transform coefficients using a lightweight stream algorithm," *Optik – International Journal for Light and Electron Optics*, vol. 124, no. 18, pp. 3693–3700, Sep. 2013.

[19] L. Mou, T. Huang, L. Huo, W. Li, W. Gao, and X. Chen, "A secure media streaming mechanism combining encryption, authentication, and transcoding," *Signal Processing: Image Communication*, vol. 24, no. 10, pp. 825–833, Nov. 2009.

[20] L. Xiao, O. Bastani, and I.-L. Yen, "An efficient homomorphic encryption protocol for multi-user systems," *IACR Cryptology ePrint Archive 2012*, pp. 193–212, 2012.

[21] C. Gentry and S. Halevi, "Fully homomorphic encryption without squashing using Depth-3 arithmetic circuits," *Proceedings of the IEEE 52nd Annual Symposium on Foundations of Computer Science*, Oct. 2011, pp. 107–109.

[22] D. Boneh, "Evaluating 2-DNF Formulas on Ciphertexts," *Second Theory of Cryptography Conference, TCC 2005, Cambridge Proceedings*, 2005, pp. 325–341.

[23] M. Van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2010, pp. 24–43.

[24] M. Tibouchi, "Batch fully homomorphic encryption over the integers," *Lecture Notes in Computer Science*, vol. 7881, pp. 315–355, 2013.

[25] C. P. Gupta and I. Sharma, "A fully homomorphic encryption scheme with symmetric keys with application to private data processing in clouds," *Proceedings of the 4th International Conference on the Network of the Future, NoF 2013*, 2013, pp. 1–4.

[26] G. Ateniese, "Verifiable encryption of digital signatures and applications," *ACM Transactions on Information and System Security*, vol. 7, no. 1, pp. 1–20, Feb. 2004.
[27] H. Ba, W. Heinzelman, C. A. Janssen, and J. Shi, "Mobile computing – a green computing resource," *Proceedings of the IEEE Wireless Communications and Networking Conference, WCNC*, 2013, pp. 4451–4456.
[28] H. Qi and A. Gani, "Research on mobile cloud computing : review, trend and perspectives," *Proceedings of the Second International Conference on Digital Information and Communication Technology and it's Applications (DICTAP),* 2012, pp. 195–202.
[29] N. Fernando, S. W. Loke, and W. Rahayu, "Mobile cloud computing: A survey," *Future Generation Computer Systems*, vol. 29, no. 1, pp. 84–106, 2013.
[30] M. Louk, "Homomorphic encryption in mobile multi cloud computing," *Proceedings of the International Conference on Information Networking (ICOIN)*, 2015, pp. 493–497.
[31] X. Fan, J. Cao, and H. Mao, "A survey of mobile cloud computing," *ZTE Corporation*, vol. 16, no. 1, pp. 393–413, 2011.
[32] M. Sookhak, F. R. Yu, M. Khurram, and Y. Xiang, "Attribute-based data access control in mobile cloud computing : Taxonomy and open issues," *Future Generation Computer Systems*, vol. 72, pp. 273–287, 2016.

Mohd Rizuan Baharon *received the PhD degree in Computer Science from Liverpool John Moores University, Liverpool, United Kingdom, in 2017. He completed his master degree in Mathematics in 2006 and his undergraduate studies in 2004 at Universiti Teknologi Malaysia, Malaysia. Currently, he is a Senior Lecturer at the Department of Computer System and Communication, Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka, Malaysia. He started his career as a lecturer at this university since June 2006. He has vast experiences in teaching Computer Science, Cryptography and Mathematics subjects. His research interests are mainly in the area of Mobile Network Security, Cloud Computing Security, Data Privacy and Integrity, Mobile Users Accountability and Cryptography. He is a lifetime member of Mathematical and Sciences Association Malaysia (PERSAMA, Malaysia). He has produced a number of journal and conference papers at national and international levels.*

Mohd Faizal Abdollah *graduated from Universiti Utara Malaysia for his first honorary degree and second degree in Information technology from Universiti Kebangsaan Malaysia. He graduated his PhD in Network Security from Universiti Teknikal Malaysia Melaka, Malaysia. Currently, he is an Associate Professor in Network Security at Universiti Teknikal Malaysia Melaka, Malaysia. He has vast experiences in teaching Computer Science and Network Security subjects. His research interests are mainly in the area of Network Security.*

Nur Azman Abu *graduated from Purdue University, West Lafayette, Indiana for his first honorary degree in Statistics and second degree Mathematics. He graduated his PhD in Cryptography from Universiti Teknikal Malaysia Melaka, Malaysia. Currently, he is an Associate Professor in Cryptography at Universiti Teknikal Malaysia Melaka, Malaysia. He has vast experiences in teaching Computer Science, Cryptography and Mathematics subjects. His research interests are mainly in the area of Cryptography, Audio and Image Processing.*

Zaheera Zainal Abidin *graduated from University of Canberra, Australia for her first degree in Information technology and second degree in Computer Networking from Universiti Teknologi Mara, Malaysia. She graduated her PhD in I.T. Quantitative Sciences from Universiti Teknologi Mara, Malaysia. Currently, she is a senior lecturer at Universiti Teknikal Malaysia Melaka, Malaysia. She has vast experiences in teaching Computer Science and Network Security subjects. Her research interests are mainly in the area of Information Security, Physical Security, Computer Networking, Image Processing.*

Ariff Idris *graduated from Universiti Teknologi MARA, Malaysia for his first honorary degree in Information Technology. He received his second degree in Information technology from Universiti Kebangsaan Malaysia. Currently, he is senior lecturer in Computer System and Communication at the Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka, Malaysia. He has vast experiences in teaching Computer Science subjects. His research interests are mainly in the area of Wireless Technology, Networking and Data Communication.*