THE PUBLISHING HOUSE
OF THE ROMANIAN ACADEMY

PROCEEDINGS OF THE ROMANIAN ACADEMY, Series A,
Volume 18, Special Issue 2017, pp. 371-380

PRIVACY PRESERVING VECTOR QUANTIZATION BASED SPEAKER RECOGNITION SYSTEM

Andrei ENE[*], Mihai TOGAN[*,**], Stefan-Adrian TOMA[*]

[*]Military Technical Academy, Computer Science Dept., Bucharest, ROMANIA
[**]certSIGN, Research and Development Dept., Bucharest, ROMANIA
E-mail: `mihai.togan@gmail.com`

Abstract. In the recent years we have witnessed an emergence of cloud computing services, in all important fields. These services provide space for keeping huge amount of data, and also great computational power. But together with these huge benefits, some privacy concerns are raised, about how much trust could we have in the provided services. A solution to this problem comes from the perspective of using homomorphic encryption to store and process data in an encrypted form. In this paper, we present the implementation of a speaker recognition system, which preserves the privacy of the K known speaker models used for identification. Our approach uses for homomorphic operations the homomorphic library SEAL and a larger plaintext space, avoiding the large amount of ciphertexts caused by bit-wise encryption.

Key words: speaker recognition, homomorphic encryption, SEAL library.

1. INTRODUCTION

The use of biometrics for personal recognition has been proved efficient and practical. However, it raises several privacy issues [1, 2]. In particular, biometrics cannot be considered as any other secret data. Biometric data are almost not revocable due to their permanent nature, whereas they are unique and can be used to identify someone among a large set of individuals. If biometric templates are stolen, they may be used for illegal activities such as identity theft or tracking individuals. The biometric identification services raise major issues concerning the privacy of user's personal information if the recognition process is performed on centralized and untrusted servers. The biometric characteristics are elements with strong personal character, and their sharing must be done carefully and in maximum security conditions. On the other hand, the possibility of outsourcing computation to the cloud offers substantial cost-savings for businesses and individuals, flexibility, and availability of computational resources, but potentially sacrifices the privacy. A solution for eliminating the risks inherent in storing and computing on sensitive data, is encryption, which makes the data unavailable without the unlocking key. The traditional model based on standard cryptographic algorithms such as AES or RSA do not allow in-cloud processing capabilities. These algorithms encrypt data in a way that makes computing on the data in its encrypted form impossible. The common operations such as searching, comparing or modifying the data cannot be performed until the data are transported back to the clients' systems. This approach leads to a decrease in scalability and flexibility, and limits the cloud infrastructure only to a data storage service. The advantage of cloud migration, namely the large computing power provided to the customers, cannot be used in this case.

Recent research in the field has proposed new cryptographic mechanisms for data protection. Privacy homomorphism is an important notion for encrypting clear data while allowing one to carry out operations on encrypted data without decryption. The homomorphic encryption (HE) can help address the problem of data privacy by allowing the user to upload encrypted data to the cloud, on which the cloud can then operate without having the secret key. Use of homomorphic encryption for data protection allows performing meaningful computations directly on encrypted data whose results are also in encrypted format. These computations can be performed without the need to previously decrypt the data or to access the keys. The cloud can return encrypted outputs of computations to the user without decrypting the data, thus providing data hosting and computing services without compromising privacy. The servers from the cloud have to know only the processing algorithm. Even the concept was first introduced by Rivest et al. in [3], the scenario we are speaking has been proven to be possible with the first fully-homomorphic encryption (FHE) scheme proposed by C. Gentry in [4].

The scheme from [4] allows performing an unlimited number of two basic mathematical operations (addition and multiplication) on encrypted data while keeping the correct result as if the operations were performed on unencrypted data. Using such a scheme, one can implement any processing circuit (algorithm) on the encrypted data. Since 2009, a lot of research effort was focused on finding better solutions that meet reasonable requirements in terms of efficiency. In the last years, there were many continuations to Gentry's work [5–10].

In many papers the data is encrypted bit-wise, which means that a ciphertext will be produced for each bit of the message. The later computation on data is done using boolean circuits with XOR and AND gates, realized as additions and multiplications modulo 2. Fortunately most schemes also allow for a larger message space. Given the fact that in practice, the most commonly used operations are addition and multiplication of integer or real numbers, it is obvious that encrypting the values using a larger message space is more efficient than decomposing values into bits and applying bit-wise operations.

Speaker recognition is used in solving two related problems: speaker identification and verification. Speaker identification tries to recognize an unknown speaker from a set of K known speakers while speaker verification tries to accept or reject a given identity claim. This paper focuses on speaker identification task. An identification system receives as input a sampled speech data and outputs the index of the identified speaker. The most important components of such a system are: feature extraction, the speaker models and the matching algorithm.

There are a few articles in literature, presenting the idea of privacy preserving biometrics. One such example can be found in [18], where the authors propose the use of additively homomorphic Paillier cryptosystem for the implementation of a privacy preserving fingerprint matching system, making use in their implementation of garbled circuits [19]. They also use for the distance measurement the Squared Euclidian Distance, for which they propose an optimization protocol, through secure two-party computation. There is also a proposal for a privacy preserving speaker verification and identification system in [20], which relays on Gaussian Mixture Models.

The authors propose a set of protocols for both speaker verification and identification. For the identification process, there are two protocols proposed. One assumes that the client sends the encrypted speech samples to the servers while the other assumes that the server will send the encrypted models to the client. The encryption is done using the additively homomorphic Paillier cryptosystem.

A privacy preserving biometric protocol which relays on homomorphic encryption was proposed in [25]. It is constructed using Goldwasser-Micali cryptosystem. The protocol rely on a distributed architecture composed from the user, the authentication server, the database and the matcher. In [26] the authors presented a possible attack on the proposed protocol, followed up by some countermeasures to improve the protocol.

A recent approach for secure biometric authentication using iris detection can be found in [24]. It uses as a metric for distance, the Hamming distance and combines a somewhat homomorphic scheme with a message authentication code (MAC) for integrity purposes and with a few defined protocols. The tests were performed using HELib, making use of the batching technique for packing several biometric features into one ciphertext.

In this paper we used homomorphic encryption in a speaker identification system. As such, we constructed a basic speaker recognition system, based on vector quantization (VQ) of mel-frequency cepstral coefficients (MFCC).

The state of the art speaker recognition systems use universal background models (UBM) based on Gaussian mixture models (GMM) [28], Gaussian mixture models and hidden Markov models (GMM+HMM) [29] or deep neural networks (DNN) [30,31]. However, for the purpose of this paper the VQ based speaker recognition was considered appropriate.

The rest of the paper is organized as follows. In **Sections 2** we briefly present the SEAL library that has been used for our implementation. **Section 3** makes several considerations on the features extraction process.

In **Section 4** we describe the details for our implementation, including the homomorphic model for the privacy-preserving speaker recognition system along with the experimental results. Finally, **Section 5** outlines our conclusions.

2. SEAL LIBRARY

Fan-Vercauteren (FV) homomorphic scheme [12] is constructed on a variant of the so called "learning with errors" (LWE) problem first introduced by Regev in [16], mainly its rings version RLWE [17]. The RLWE problem is to distinguish between the distribution $A_{s,\chi}^{(q)}$ and the distribution $U(R_q^2)$, where $A_{s,\chi}^{(q)}$ denotes the distribution obtained by choosing a uniformly random element $a \leftarrow R_q$, a noise term $e \leftarrow \chi$ and outputting $(a,[a \cdot s + e]_q)$ and $R = Z[x]/(f(x))$, with $f(x)$ a cyclotomic polynomial $\Phi_m(x)$. The RLWE problem can be reduced to the shortest vector problem over ideal lattices [17]. Furthermore, s can be sampled from χ instead of R_q without any security implications.

The plaintext space is taken as R_t for some integer $t > 1$. Let $\Delta = \lfloor q/t \rfloor$ and $r_t(q) = q \bmod t$. Integers t and q do not have to be prime, nor t and q coprime. For the encryption, a secret key sk is sampled from the distribution χ. The public key will be $pk = ([-(a \cdot s + e)]_q, a)$, where s is the secret key, $a \leftarrow R_q$ and $e \leftarrow \chi$. To encrypt a message $m \in R_t$, let $p_0 = pk[0]$, $p_1 = pk[1]$ and $u, e1, e2 \leftarrow \chi$. Then the ciphertext ct representing the encryption of message m is: $ct = ([p_0 \cdot u + e_1 + \Delta \cdot m]_q, [p_1 \cdot u + e_2]_q)$. To recover the message m, the decription will be applied as follows: let $c_0 = c[0]$, $c_1 = c[1]$ and s the secret key. Then the computation of: , $[\lfloor \frac{t \cdot [c_0 + c_1 \cdot s]_q}{q} \rceil]_t$, will recover the message m, while the noise present in ciphertext is less than $\frac{\Delta}{2}$.

The simple arithmetic encryption library (SEAL) [11], developed by Microsoft research group, is a library written in C++ for making computations on encrypted data. It is built over the FV scheme.

The plaintext space is $R_t = Z_t[x]/(x^n + 1)$, polynomials of degree less than n with coefficients modulo t, t representing the plaintext modulus. For encrypting a number (integer or real), it first needs to be encoded into a plaintext polynomial in R_t, and afterwards encrypted. The ciphertext space is $R_q = Z_q[x]/(x^n + 1)$, with q being the coefficient moduls.

The encryption parameters are: the degree n, the moduli q and t, the decomposition word size w and two distributions χ_{key}, χ_{err}.

The parameter n is the maximum number of terms in the polynomials, used to represent plaintexts and ciphertexts.

In SEAL, n is always chosen as a power of 2. The polynomial $X^n + 1$ is called polynomial modulus, denoted as *poly_modulus* in SEAL. The parameter q is the coefficient modulus used to reduce the coefficients of ciphertext polynomials. It is called *coeff_modulus* in SEAL. The parameter t is an integer modulus, used for plaintext coefficient reduction, called *plain_modulus*. The integer w is the base to which decomposition of integer coefficients in smaller parts is done. In practice w is a power of 2, and $log_2(w)$ is the decomposition bit count.

Any element $a \in R$ can be written as $a = \sum_{i=0}^{n-1}(a_i x^I)$, with $a_i \in Z$. All plaintext, ciphertexts, encryption and decryption keys are elements of ring R and have this form.

Homomorphic additions are performed by computing component-wise sum of the arrays, while multiplication is simply polynomial multiplication modulo $X^n + 1$.

The library offers for the encoding process some pre-defined encoders for different needs: *BinaryEncoder*, *FractionalEncoder*, *IntegerEncoder* [11]. The *IntegerEncoder* routine encodes integers as plaintext polynomials in the following way: a base b, expansion of the integer is computed. If no value is given to the constructor, the default $b = 2$ is used.

The polynomials have coefficients modulo b, integers between $-(b - 1)/2$ and $(b - 1)/2$ when b is odd and when it is even the range is between $(b/2)$ and $(b - 1)/2$. For example the integer $x = 26$, for a base $b = 2$, will be encoded as the polynomial $x^4 + x^3 + x^1$.

In case of *FractionalEncoder*, the integer part is represented the same way as an integer, but the fractional part will be represented in the leading coefficients of the polynomial, with inverted sign. The negative coefficients will be represented as their negatives modulo plain modulus.

3. FEATURES EXTRACTION

In speech recognition, speaker recognition, or image processing, the feature extraction is used to build non-redundant yet informative values from an initial set of measured data. The number of further computation is reduced, only a small set of coefficients being used instead of the whole set of data.

There are many types of acoustic features that can be used in speaker recognition applications, such as mel-frequency cepstral coefficients [32], perceptual linear prediction coefficients [34], voice source (mel-frequency) cepstrum coefficients [27] and so on. However, in speech and speaker recognition systems, most commonly used features are MFCCs [33].

MFCCs are extracted from recorded speech, processed through a pre-emphasis filter. The signal is split into overlapping fixed length frames - usually 20 - 40 ms frames with 10 ms overlap. A weighting window is applied to each frame, in this case a Hamming window. The frames are then transformed to the frequency domain, and each one is processed through a mel-filter bank. The next step is applying a logarithmic function to each frame, and then transforming back to the time domain. For each processed frame X, a fixed length feature vector $C = \{c_1, \cdots ,c_k\}$ will be computed and recorded, describing the acoustic particularities of that frame. The feature vector can be enhanced with the first and second derivative of the MFCCs, pitch information (useful for discriminating male speakers form female speakers), frame energy its first and second derivative. However, increasing the dimensions of the feature vector increases also the requirements of the computational resources.

4. SPEAKER RECOGNITION SYSTEM

We choose to implement the vector quantization based speaker recognition system presented in [13], but enhanced with privacy preserving capabilities. The model of the speaker recognition system is depicted in Figure 1. The enrollment module is used for creating a mathematical model of the speaker, while the recognition module is used for identifying the speaker. After enrollment, the generated speaker models are encrypted using homomorphic methods and then stored in cloud. For speaker identification, spectral features are extracted, encrypted and send for verification at the server on cloud, where matching in the homomorphic domain is performed.

In both modules pointed in Figure 1, the first step consists in the feature extraction. From each speech signal, a set of features will be extracted into a vector $X = \{x_1, \cdots ,x_L\}$. The feature extraction algorithm divides the signal into fixed length frames, and then extract the MFCC coefficients (we discard first coefficient and keep the other 12 for each frame). Afterwards it is applied Linde-Buzo-Gray vector quantization algorithm [21] on the extracted feature vectors, for locating the clusters.

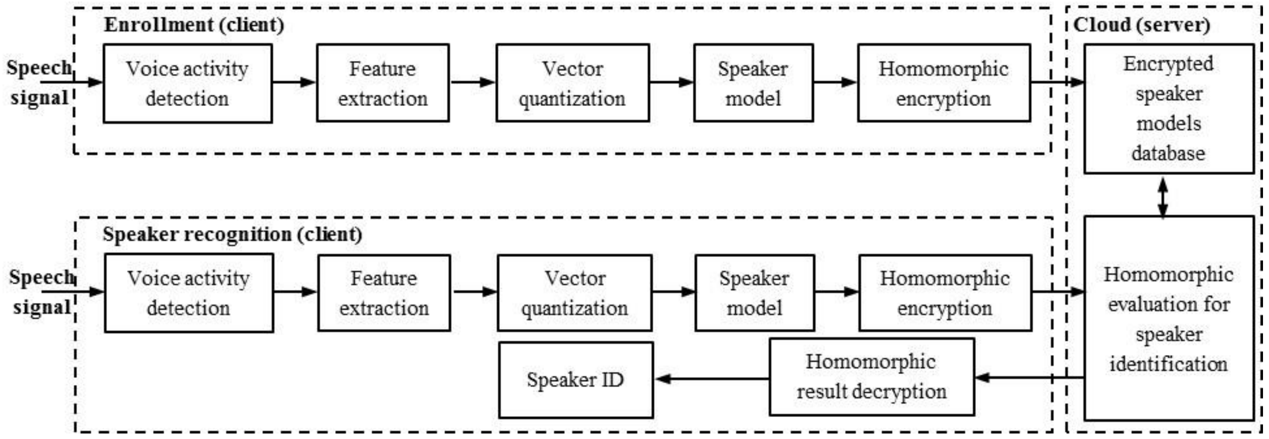

Fig. 1 – The model of the speaker recognition system.

We give as input our feature vectors and receive a codebook C containing the cluster centroids. According to our privacy criteria, the server cannot observe the speech data belonging to the client, nor can the client observe the speaker models belonging to the server. To achieve privacy criteria for the speaker models held by the servers, we homomorphically encrypt the codebooks C, for every speech signal, and then store them in the database.

We propose in our implementation the encryption of the values, representing our codebooks with centroid vectors, as real numbers. It is obvious that this approach is more efficient than bit-wise encryption. Decomposing every coefficient value into bits and then build circuits to homomorphically evaluate addition or multiplication would represent a major drawback compared to using a larger plaintext space.

Given a set of homomorphically encrypted feature vectors from an unknown speaker for identification $X = \{x_1,\cdots,x_L\}$, thus achieving our second privacy criteria for the data belonging to the client, we compute for each Codebook C the distortion $D_i = d(X,C_i)$ between X and C_i. The identification of the unknown speaker is done with the codebook with which our feature vectors X gets the smallest distortion. The distortion measure d approximates the dissimilarity between the codebook $C_i = \{c_{i1},c_{i2},\cdots,c_{iK}\}$ and our unknown feature vectors X. We map each vector in X to the nearest code vector from C_i and compute the average of these distances:

$$d(X,C_i) = \frac{1}{L}\sum_{j=1}^{L} \min_{k=1,K} d_{ES}(x_j,c_{ik}) \tag{1}$$

where d_{ES} is the Squared Euclidian distance:

$$d_{ES}(x,y) = \sum_{i=1}^{dim}(x_i - y_i)^2 \tag{2}$$

Even though the distance returned is not correct as a metric (because it does not satisfy triangle inequality), it is suitable and often used as an optimization method, when there are only distance comparisons needed.

For evaluating the minimum between two encrypted real numbers, we propose using the Quadratic Formula for finding minimum of two numbers [6]:

$$\min(r,s) = \frac{1}{2}(r+s+|r-s|) \tag{3}$$

After this process, we will obtain an array $D = d_0, d_1,...d_K$ of the encrypted distortion measurements, between the signal received for identification and the K known speakers from our database. Knowing that the identification is done with the smallest distortion measurement, this means that our identified speaker will be represented by the index of the minimum value from our array D. To obtain the index of our speaker we must define a privacy preserving protocol for finding the nearest neighbor (NN) [22]. We will first define a protocol for extracting the minimum value from an array of encrypted real numbers ($t > 2$). Then we propose a secure way to find the NN from the array containing distortion measurements. Then, we propose a method to extract a minimum value, without the use of the protocol, even though it is hardly implementable at this time.

4.1. Minimum Problem

Let's construct a protocol to find the minimum value between two real numbers. We consider r and s two homomorpichally encrypted real numbers. Then we call F, an evaluation function that return the minimum value between them, hence: $F(r, s) = Enc(\min(r, s))$. Applying the mentioned function F, on a vector of values $V = v_1, v_2,...,v_n$, extends our protocol, allowing us to extract the minimum value from an encrypted vector. Hence $F(r, s) = Enc(\min(r, s))$, it means that for our vector V, a value v_i is the minimum if the following relation holds:

$$\begin{gathered}
v_i = \min(v_i, v_1) \Leftrightarrow F(v_i, v_1) = \tfrac{1}{2}(v_i + v_1 - |v_i - v_1|) \\
v_i = \min(v_i, v_2) \Leftrightarrow F(v_i, v_2) = \tfrac{1}{2}(v_i + v_2 - |v_i - v_2|) \\
\vdots \\
v_i = \min(v_i, v_n) \Leftrightarrow F(v_i, v_n) = \tfrac{1}{2}(v_i + v_n - |v_i - v_n|)
\end{gathered}$$

To apply the quadratic formula, we need to always obtain the absolute value of the difference of the two numbers we want to compare. Let $z = Enc(r - s)$. Having the difference z encrypted, we define a protocol for sending the difference back to the client for decryption. The client decrypts z and sends back to server an encryption of its absolute value $Enc(|z|)$.

4.2. Privacy preserving nearest neighbor

Given an encrypted array of numbers, selecting the nearest neighbor involves finding the minimum value from our array and then getting its corresponding data point (index). Due to our privacy criteria stated above, it is vital that the nearest neighbor finding protocol, should not reveal the index i corresponding to the minimum value, to the server, nor should it reveal the array to the client. Given the fact that we can extract the minimum value from our array D, using the protocol introduced above, we will call it $m = \min_{i=0,K}(d_i)$.

We successfully obtained the minimum value, but we do not know to which speaker from our database it belongs to. To find the nearest neighbor we were inspired by the work in [22] [23], even though their methods could not be used in our proposal (applying bit decomposition on our ciphertext was not possible). Thus knowing that we are able to extract the minimum value, we define the following protocol for finding the nearest neighbor it belongs to:

1. Server has the encrypted values $D = \{d_0, d_1, \ldots, d_K\}$.
2. Server work with the client, through defined minimum protocol (4.1), to get the minimum value $m = \min_{i=0}^{K}(d_i)$.
3. Server computes $D' = \{(d'_0, d'_1, \ldots, d'_K\}$ where $d'_i = m - d_i$.
4. Server send the array D' to the client for decryption.
5. Client decrypts D' and get the index of the 0 value, representing the index of the smallest value from the array

During the defined protocol, no information has been revealed neither to the client or the server. Even though the privacy is granted, the client should be unable to recover the original values from the array he receives, to achieve an even better privacy protection, the server could multiply each difference from D' with a constant chosen from some small random interval. If this is taken into account, additional care must be taken, to ensure that the homomorphic multiplication with a constant does not exceed the noise in the final ciphertext above some level making it impossible to decrypt.

4.3. Calculating the minimum without protocol

The major drawback in our first proposal for finding the minimum from a vector, is the number of exchanges between the client and the server. This comes from the theoretic impossibility of finding the absolute value of an encrypted value. We present in this section an idea of how this problem could be solved, resulting in a secure way of obtaining the minimum value, between two integers without any data transfer between client and server being needed in the process.

Our idea is based on the bootstrap technique first introduced by Gentry in [4]. In order to make a encryption scheme fully homomorphic, it needs to be able to control the growth of the noise present in ciphertext, maintaining it below a certain level, so the decryption circuit remains valid, even for an unlimited number of homomorphic operations.

To refresh a ciphertext c_i, which is the encryption of the plaintext π with a public key pk_i, the ciphertext will be reencrypted with a public key pk_{i+1}, and the decryption circuit is homomorphically applied. In this way it is obtained the encryption of the plaintext π under the public key pk_{i+1}. Doing so, the ciphertext is decrypted, but the plaintext is never revealed, being covered all the time by at least one layer of encryption.

Our idea uses the same technique, but is not meant to refresh the noise in ciphertext. It should help us obtain the absolute value of an encrypted integer. Assume a given ciphertext z, which represents the difference between the two numbers we want to extract minimum from. Finding the absolute value problem is reduced upon finding if our ciphertext's value is greater than 0.

The way SEAL encodes an integer into a plaintext polynomial (it was explained in Section 2), is helpful for achieving our goal. A positive integer has its plaintext coefficients equal to 1, while a coefficient from the encoding of negative integers has a value of -1 modulo *plaintext_modulus*. Thus we propose the following algorithm for finding the absolute value of the encrypted integer z:

Let (pk_1, sk_1), the pair of keys used for encrypting our ciphertext z, and (pk_2, sk_2), a pair of keys used for evaluation. Let's call $F(pk_2,D,\bar{z},\overline{sk},M)$ an evaluation function that allows us to verify if 0 is greater than our ciphertext z, where $\bar{z}$ is the encryption of our ciphertext under the public key pk_2, $\overline{sk}$ is the encryption of the first secret key sk_1 under pk_2 and M is the encryption of the *plain_modulus*: $M = Enc(pk2, plain\_modulus)$. Knowing that a FHE scheme, must be able to evaluate its own decryption circuit, we define it as D and this means that we can decrypt our variable z, under a layer of encryption with pk_2.

$$Set\,\Psi \leftarrow -Eval(pk_2,D,\bar{z},\overline{sk}) \tag{4}$$

The function *Eval* returns an encryption, Ψ, under pk_2 of one of the coefficients of the plaintext polynomial, meaning $D(\Psi,sk_2) = D(z,sk_1)[i]$, where i is the index of a coefficient of the plaintext polynomial $D(z,sk_1)$. Finally the result returned by our function F will be $Y = \Psi - M$, hence:

$$F(pk_2, D, \bar{z}, \overline{sk}, M)) = \begin{cases} Enc(-(M-1), pk_2) = Enc(1, pk_2) mod M, & z > 0 \\ Enc(M-1, pk_2) = Enc(-1, pk_2) mod M, & z < 0 \end{cases}$$

Multiplying our initial z with the result Y, obtained by function F modulo *plain modulus* will always return the absolute value of z: $|z| = Y * z \ mod\ M$. In order for this multiplication to be possible, and for making further computations with the absolute value of z, the bootstrapping technique must be applied on its initial ciphertext and also on all other variables implied in the equation, in order to keep all ciphertexts encrypted under the same public key pk_2. Having the absolute value of z, we can now easily compute the quadratic formula (3) for extracting the minimum between the two integers.

The idea above can be implemented using any FHE scheme, or any library, as long as the way of encoding integers into polynomials through binary expansion remains unchanged. We were not able to fully implement it using SEAL, due to the fact that the bootstrapping technique is not implemented in the library. Using this approach for minimum problem in our speaker recognition system, would imply converting the signal samples from real values to integers. This could be achieved encoding them into 8 bits integers, in order to be smaller than our plaintext modulus.

4.4. Experimenting with SEAL

Our experiments with SEAL were conducted on a machine equipped with an Intel I7 2670 QM at 2.10 GHz CPU and 8GB memory. We implemented and tested the speaker identification algorithm described in Section 4 using the protocol proposed in Section 4.1 to find the minimum value between two real numbers. The parameters in SEAL were set as follows: *Poly_modulus* $x^{1024} + 1$, $x^{2048} + 1$ and $x^{4096} + 1$, *plaintext_modulus* 2^5, using default parameter options for *coeff_modulus* at 2048 and 4096. In SEAL, each multiplication increases the noise by a huge factor and each homomorphic addition, sums up the noises of the operands. For the computation of Squared Euclidian Distance, the difference is squared and then summed with the others, resulting a ciphertext with big noise. Further computation done for extracting the distortion measurement from the distance vectors also increase the noise present in ciphertext, resulting in a final ciphertext for the distortion measurement for which decryption will fail to retrieve the correct plaintext. Using the noise estimations described in [11] section 6.1, we concluded that for our implementation, the calculus of the distortion measurement will retrieve a correct value if only the coefficient modulus is greater than or equal than the recommended value for $n = 2048$, $coeff\_modulus \geq 2^{60} - 2^{14} + 1$, on 60 bits.

The strength of this library is its easy configurable parameters compared with other existing libraries as developers admit in [11]. The polynomial modulus we choose drastically affects the performance of our application. But the problem comes because, the security of the scheme is granted by the parameters n, q and σ. Knowing that in SEAL, standard noise deviation, σ is equal to 3.19, the security of the scheme is affected by the choice of n and q, more precisely by the difference between them. We chose for testing, besides the recommended parameters for $n \geq 2048$, a custom configuration: $n = 1024$, with a $q = 2^{60}-2^{14}+1$, to maintain our distortion's measurement ciphertext valid for decryption. We tested the security of the scheme using the Sage module LWE-Estimator described in [14], to estimate the hardness of concrete LWE instances. Our parameter choice lead to an estimate cost of 56 bits of security, while the authors obtained for the parameters recommended by SEAL for n = 1024, an estimate cost of 68 bits of security in [15].

For the training phase we used 5 or 3 (for higher polynomial modulus) speech signals, recorded in a laboratory environment, where every speaker read the same word. For each speaker, we recorded two utterances, one for the training phase and one for recognition. We used for feature extraction and vector quantization Matlab 2017. We performed short-term melcepstrum analysis with a 20 ms Hamming-window, with 10 ms shift. The number of mel-cepstral coefficients (dimension of feature vectors) was selected as 12, and the first coefficient c_0 was discarded as usually. We choose for the quantization algorithm 8 cluster vectors for each codebook C_i. The result is a codebook for each speaker, made up from 8 (distinct) points in the vector space of the MFCCs. The reason we choose to use only 8 cluster vectors for each codebook was to reduce the number of computation needed for calculating the Euclidian distance as much as possible.

To test the accuracy of our speaker recognition system, we used our custom setup for the library, because it provides the smallest execution time, and a database consisting in five speakers. We used in the identification process the second recording of one of the speakers from the database and repeated the process five times, trying to obtain an identification with the second recorded speech signal for all our known speakers. Our system succeeded to correctly identify 4 of our 5 unknown speakers. The encryption of data and processing it in its encrypted form was done using SEAL library v2.1, built under Microsoft Visual Studio 2015 on Windows 10 Pro.

Table 1

The time and memory costs required for evaluation of the speaker recognition system

Poly modulus	coeff modulus	Total Time	Time for one speaker	Enc Time	Memory	Speakers
$X^{1024} + 1$	60 bits	4140 s	726 s	110 s	160 Mb	5
$X^{2048} + 1$	60 bits	6598 s	2283 s	348 s	290 Mb	3
$X^{4096} + 1$	116 bits	7925 s	2789 s	680 s	566 Mb	3

Table 1 presents the execution times and the average memory needed (the arrays of the encrypted real numbers are loaded entirely into the memory) for our speaker recognition implementation using the two setups for the library. The table also contains the time required for encryption (Enc Time) of our codebooks C_i summed with the time required for encryption of the feature vectors of the test speech signal X.

5. CONCLUSIONS

In this paper we proposed an implementation of a speaker recognition system which preserves privacy of the K known speakers used for identification. Our solution is an implementation of the vector quantization algorithm using real numbers encrypted in a large plaintext space. The focus of the paper was to show that a privacy preserving speaker recognition system can be constructed, using fully homomorphic encryption. The system uses a simple speaker recognition algorithm, based on the vector quantization of Mel-frequency cepstral coefficients. A thorough testing of the speaker recognition algorithm was beyond the scope of this paper. However, other authors reported accuracy results as high as 95% [36], depending on test data and setup, without encryption. A proper testing of the system will be conducted in future work, we will use a speaker database with a large number of speakers and test the algorithm for both encrypted and clear processing. We intend to test also homomorphic encryption on state of the art speaker recognition system. Current state of the art speaker recognition systems are based on *i*-vectors [35]. Hence, future developments of our fully homomorphic speaker recognition system will see the inclusion of i-vector based recognition. Because of the fact that the numbers are not bit-wise encrypted, we had to create a protocol to obtain the minimum value between two encrypted numbers. Therefor a communication with the client is constantly required. It is clearly visible that our proposed method for encrypting, requires a big amount of data to be transferred between client and server in order for obtaining through our proposed protocol the desired values. But our strength comes from the fact, that the memory used for this application is significantly reduced, even though, there are some big vectors of ciphertexts held in memory during the whole process. We also proposed an idea for finding the minimum value between two encrypted numbers, using the same formula. But instead of using the protocol for retrieving the absolute value, we propose a theoretical method which will return the absolute value for an encrypted integer, based on bootstrapping idea.

ACKNOWLEDGEMENTS

This work was supported by the Romanian National Authority for Scientific Research (CNCSUEFISCDI) under the project PN-III-P2-2.1-PTE-2016-0191 (ctr. no. 54PTE/2016).

REFERENCES

1. S. PRABHAKAR, S. PANKANTI, A.K. JAIN, *Biometric recognition: Security and privacy concerns*, IEEE Sec. Privacy, **1**, 2, pp. 33–42, 2003.
2. N. K. RATHA, D. ZHANG, *Privacy protection in high security biometrics applications*, ICEB, **6005**, pp. 62–69, Springer, 2010.
3. R. RIVEST, L. ADLEMAN, M. DERTOUZOS, *On Data Banks And Privacy Homomorphisms*, Foundations of Secure Computation, **4**, *11*, pp. 169–180, 1978.
4. C. GENTRY, *A Fully Homomorphic Encryption Scheme*, PhD Thesis, Stanford University, http://crypto. stanford.edu/craig, 2009.
5. M.V. DIJK ET AL., *textitFully Homomorphic Encryption Over The Integers*, Advances in Cryptology, Eurocrypt 2010, Lecture Notes in Computer Science, pp. 2443, 2010.
6. N.P. SMART, F. VERCAUTEREN, *Fully Homomorphic SIMD Operations*, Cryptology ePrint Archive 2011/133.
7. Z. BRAKERSKI, C. GENTRY, V. VAIKUNTANATHAN, *Fully Homomorphic Encryption without Bootstrapping*, Innovations in Theoretical Computer Science Conference, pp. 309325, 2012.
8. T. GRAEPEL, K. LAUTER, M. NAEHRIG, *ML Confidential: Machine Learning on Encrypted Data*, Information Security and Cryptology, LNCS, pp. 121, 2013.
9. J.-S. CORON, T. LEPOINT, M. TIBOUCHI, *Batch fully homomorphic encryption over the integers*, IACR Cryptology ePrint Archive, 2013.
10. J. H. CHEON, M. KIM, M. KIM, *Search-and-compute on Encrypted Data*, IACR Cryptology ePrint Archive, 2015.
11. H. CHEN, K. LAINE, R. PLAYER, *Simple Encrypted Arithmetic Library - SEAL v2:1*, IACR Cryptology ePrint Archive, 2017.
12. J. FAN, F. VERCAUTEREN, *Somewhat Practical Fully Homomorphic Encryption*, IACR Cryptology ePrint Archive, 2012.
13. T. KINNUNEN, T. KILPELINEN, P. FRNTI, *Comparison of clustering algorithms in speaker identification*, 2000.
14. M. R. ALBRECHT, R. PLAYER, S. SCOTT, *On the concrete hardness of Learning with Errors*, IACR Cryptology ePrint Archive, 2015.
15. M. R. ALBRECHT, *On dual lattice attacks against small-secret LWE and parameter choices in HElib and SEAL*, IACR Cryptology ePrint Archive, 2017.
16. O. REGEV, *On Lattices, Learning With Errors, Random Linear Codes And Cryptography*, ACM Symposium on Theory of Computing, pp. 84-93, 2005.
17. V. LYUBASHEVSKY, C. PEIKERT, O. REGEV, *On Ideal Lattices And Learning With Errors Over Rings*, Eurocrypt, Lecture Notes in Computer Science, pp. 1-23, 2010.
18. Y. HUANG, L. MALKA, D. EVANS, J. KATZ, *Efficient Privacy-Preserving Biometric Identification*, 18th Network and Distributed System Security Conference (NDSS 2011), 2011.
19. A. C. YAO, *How to Generate and Exchange Secrets*, 27th Symposium on Foundations of Computer Science, 1986.
20. M. A. PATHAK, B. RAJ, *Privacy-Preserving Speaker Verification and Identification Using Gaussian Mixture Models*, IEEE Transactions On Audio, Speech, And Language Processing, VOL. **21**, NO. 2, FEBRUARY 2013.
21. Y. LINDE, A. BUZO, R. GRAY, *An Algorithm for Vector Quantizer Design*, IEEE Transactions on Communications, 1980.
22. S. RANE, P. BOUFOUNOS, *Privacy-Preserving Nearest Neighbor Methods*, IEEE Signal Processing Magazine, **30**, *2*, 2013.
23. M. J. ATALLAH, F. KERSCHBAUM, W. DU, *Secure and Private Sequence Comparisons*, WPES'03
24. J. H. CHEON, H. CHUNG, M. KIM, K. W. LEE, *Ghostshell: Secure Biometric Authentication using Integrity-based Homomorphic Evaluations*, IACR Cryptology ePrint Archive, 2016.
25. J. BRINGER, H. CHABANNE, M. IZABACHENE, D. POINTCHEVAL, Q. TANG, S. ZIMMER, *An Application of the Goldwasser-Micali Cryptosystem to Biometric Authentication*, Pieprzyk J., Ghodosi H., Dawson E. (eds) Information Security and Privacy. ACISP 2007. Lecture Notes in Computer Science, **4586**. Springer, Berlin, Heidelberg.
26. A. ABIDIN, A. MITROKOTSA, *A Privacy-Preserving Biometric Authentication Protocol Revisited*, Cryptology and Network Security. CANS 2014. Lecture Notes in Computer Science, **8813**. Springer, 2014.
27. J. GUNASON, M. BROOKES, *Voice source cepstrum coefficients for speaker identification*, IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2008), pp. 4821-4824, 2008.
28. D.A. REYNOLDS, T.F. QUATIERI, R. B. DUNN, *Speaker Verification Using Adapted Gaussian Mixture Models*, Digit. Signal Process, **10**, *1*, pp. 19-41, 2000.

29. M. K. OMAR, J. PELECANOS, *Training universal background models for speaker recognition*, The Speaker and Language Recognition Workshop (Odyssey 2010), pp. 25-57, 2010.
30. Y. LEI, N. SCHEFFER, L. FERRER, M. MCLAREN, *A novel scheme for speaker recognition using a phonetically-aware deep neural network* , IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1695-1699, 2014.
31. D. SNYDER, D. GARCIA-ROMERO, D. POVEY, *Time delay deep neural network-based universal background models for speaker recognition* , 2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), pp. 92-97, 2015.
32. S. OMID, J. PELECANOS, S. GANAPATHY, *The IBM Speaker Recognition System: Recent Advances and Error Analysis*, CoRR, **abs/1605.01635**, URL: http://arxiv.org/abs/1605.01635 [06/30/2017], 2016.
33. A. LAWSON, P. VABISHCHEVICH, M. HUGGINS, P. ARDIS, B. BATTLES, A. STAUFFER, *Survey and evaluation of acoustic features for speaker recognition* , 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5444-5447, 2011.
34. S. SHAFEE, B. ANURADHA, *Speaker identification and Spoken word recognition in noisy background using artificial neural networks* , 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), pp. 912-917, 2016.
35. M. SENOUSSAOUI, P. KENNY, N. DEHAK, P. DUMOUCHE, *An i-vector Extractor Suitable for Speaker Recognition with both Microphone and Telephone Speech*, Odyssey Speaker and Language Recognition Workshop, 2010.
36. H.B. KEKRE, V. KULKARNI, *Performance Comparison of Automatic Speaker Recognition using Vector Quantization by LBG KFCG and KMCG*, International Journal of Computer Science and Security, (IJCSS), **4**, 6, pp. 569-577, 2010.