A Time-Efficient Symmetric Key Encryption Technique for ECG Waveforms

Pooja Premnath[0000−0001−9017−4335]1, Sanjai Balajee Giridharan[0000−0003−3078−5470]2, Prahalad Rajagopal[0000−0003−2919−9722]3, Chamundeswari Arumugam[0000−0003−0200−6337] 4, and Dabbu Suman Reddy[0000−0002−6749−2498]5

[1,2,3,4]Department of Computer Science and Engineering, Sri Sivasubramaniya Nadar College of Engineering, Chennai
[5]Department of Biomedical Engineering, University College of Engineering, Osmania University, Hyderabad, Telangana

pooja2110152@ssn.edu.in

Abstract. The transmission of medical data, like Electrocardiograms (ECGs), has to be done in a systematic and secure manner. Encryption techniques, especially symmetric methods, using a single key, are simple methods to encrypt data. This paper deals with a novel method for the encryption of ECG wave forms, in the form of .wav files, using matrices. Typical methods that make use of matrices are computationally intensive, especially as the size of the file increases. Unlike other methods, this paper focuses on enhancing the overall computation process, for both encryption and decryption. The proposed algorithm drastically reduces the overall time taken, by arriving at the ideal dimensions of the matrix, to minimize the padding of zeroes, in order to make it square. This reduces the overall difficulty of generating a random key, as well as finding its inverse for decryption. Metrics are also applied to evaluate the overall performance of the algorithm.

Keywords: Electrocardiogram (ECG) · Signal Encryption · Signal Decryption ·

1 Introduction

Medical data is highly sensitive, containing private patient records and information on their health insurance. This kind of data must be encrypted, to ensure patient confidentiality and protect it from the risk of being manipulated. Transmission of such data for remote analysis must also be done securely. Health providers that do not make encryption of their patient data a priority are at risk of hefty penalties and lawsuits, and more importantly, a loss of trust of their patients. The Electrocardiogram (ECG) is an important signal, that may require to be transmitted from one healthcare center to another. In this paper, we propose a method to encrypt and decrypt ECG signals, using a simple private key encryption technique, considering the computational complexity. The performance

of the proposed algorithm is also analyzed using parameters to measure computation time and techniques to measure the similarity between the encrypted and decrypted wave forms.

An Electrocardiogram (ECG or EKG) is a tool used to record electrical signals from the heart[6]. This is used to check any abnormalities in the heartbeat, and underlying heart conditions. An ECG is developed by a nerve impulse stimuli to the heart, through electrodes. These electrodes sense the electrical activity of the heart, after placing them on different parts of the chest and the skin. The components of an ECG include the P wave, a T wave, and a QRS complex. The conventional ECG setup has 12 leads, which are of two kinds-limb leads and precordial leads. The limb leads monitor the heart along the vertical plane, while the precordial leads take readings along the horizontal plane. ECG signals are not periodic in nature. The different waves, P, T, and the QRS complex have different frequencies. The QRS has a high frequency of oscillation. The T region has lower frequencies, and the P region has even lower frequencies [3].

2 Literature Survey

Existing symmetric key encryption methods used for encrypting .wav files involve capturing the size of the original wave file, and increasing its size until it is a square number. Zeroes are padded, to make the entire matrix square and it is of double data type to maintain the values that correspond to the sample range. Then, another random double matrix of the same size is generated, that acts as the private key [2]. In the case of stereo wave files, with two channels, the sampled data from one channel is concatenated at the end of the other channel, and then the column matrix, hence generated is made a square matrix[9]. Advanced encryption techniques involve the compression of the ECG data, to facilitate transmission, which inevitably leads to compromising the quality of the signal[7]. Another method of encryption proposed to be used on ECG data involves the fusion of wavelet coefficients[1] with a random signal. This method involves using wavelet transforms to compress a given ECG signal, and then encode it using the Huffman encoding. A chaos-based encryption algorithm is also developed, wherein a combination of fusion, substitution, and chaotic permutations are used. This method of encryption features multiple processes and hence makes it much more secure[3, 11]. Symmetric block ciphers like the AES algorithm have also been applied to ECG data by performing successive contrasting transformations on data block bits iteratively. Though quite secure, the AES algorithm is found to have very high computational complexity, which while offering a high level of security to the data, also results in the unnecessary occupation of large amounts of memory[5]. Another system developed involves the encryption of vital signs, right at a Body Area Network (BAN), while the ECG is conducted[13]. This method considers that BANs utilize very low power, while complex encryption schemes like AES or ellipse curve cryptography would lead to high power consumption. The different parameters of the ECG signal are taken as parameters, to generate the initial keys. The encryption is done by

generating a stream cipher, using the Linear Feedback Shift Register (LFSR). This technique facilitates fast conversion speeds, along with the usage of simple hardware[4].

In techniques where matrix operations are used on the sampled data, the column matrix has to be made square, for any kind of matrix operations to be performed. In this scenario, zeros are padded, to all of the remaining columns, making the calculation of the product of the original sampled data and the random key, along with the inverse of the matrix, for decryption, computationally very complex, due to the large size of the matrices. In cases where the ECG has twelve leads, the length of a single concatenated column would become even larger. Rather, if the sampled data is optimized in such a fashion that the least number of zeroes have to be added to either rows or columns, generating a square matrix would be much more efficient and less time-consuming. Generating a square matrix with zeros padded to the remaining columns becomes very difficult.

3 The Proposed Symmetric Key Technique on ECG Data

This paper proposes a methodology to efficiently carry out matrix operations for private key encryption. An algorithm is developed to reshape the column matrix into another matrix, such that the number of zeroes that have to be added to the rows and columns to make a square matrix is minimal. A random key matrix with the size of the square matrix is generated and multiplied with the original data to produce an encrypted product. To decrypt the matrix, the inverse of the key matrix is computed, the padded zeroes are removed, and the matrix is then reshaped to its original two-channel format.

The paper initially presents a block diagram describing the methodology proposed. The waveform is first read, and the unencrypted data is plotted and analyzed. The next section presents the flow of the program used to implement the methodology and the pseudocode. The encrypted waveform is then plotted, and analyzed with respect to the original waveform. Then the process of decryption is elaborated on, and the decrypted values are compared with the original matrix. Most importantly, the variation in computation time is analyzed, in the scenarios where the number of zeroes padded is not optimized versus the time taken using the proposed algorithm. The level of security of the solution and the possibilities of discovering the secret key are discussed. Finally, the results of the simulation are discussed using metrics like the Mean Square Error and the Structural Similarity Index (SSM). Histograms of the encrypted and decrypted signals are also plotted to visualize the distribution of the signals.

The following block diagram shows the process of encrypting and decrypting the ECG waveform.

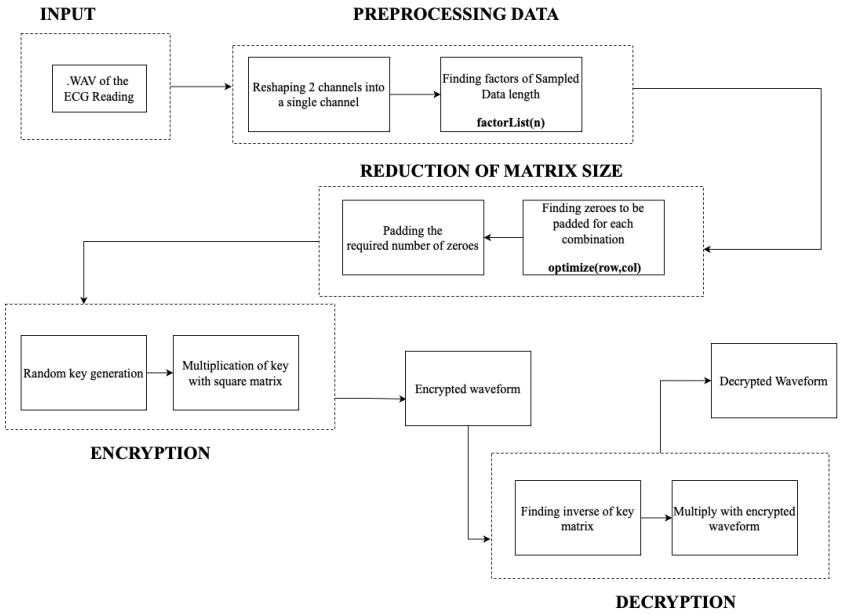

Fig. 1. Block diagram of the pre-processing, reduction of matrix size, encryption, and decryption of ECG data

4 Preprocessing the ECG Waveform

The ECG waveforms that are analyzed, encrypted, and decrypted in this paper are from the MIT-BIH Arrhythmia Database, which contains ambulatory ECG recordings, each for around thirty minutes[8]. The proposed method uses an ECG waveform in a .wav file format. The Waveform Audio File Format, stores audio in the form of bitstreams, as chunks. The .wav file is a representation of samples of sound waves that vary from the equilibrium pressure of the air[12]. The sampled data of the ECG is read, along with the sampling rate from the .wav file. This leads to almost 1300000 values of sampled values, in a concatenated column matrix, obtained from the two-channel data.

To carry out the proposed methodology, MATLAB is used to read and perform matrix operations, and Python, along with the libraries, Numpy, Pandas, and Matplotlib are used for data visualization and analysis.

The first ten seconds of the original waveform, with both channels, are shown in Figure 2.

The two-channel waveform is concatenated, to obtain a column matrix. The proposed methodology is divided into three parts: making an optimally padded square matrix, encryption, and decryption.

To encrypt the ECG waveform, the column matrix must be converted into a square matrix, to perform matrix operations, such as the computation of the

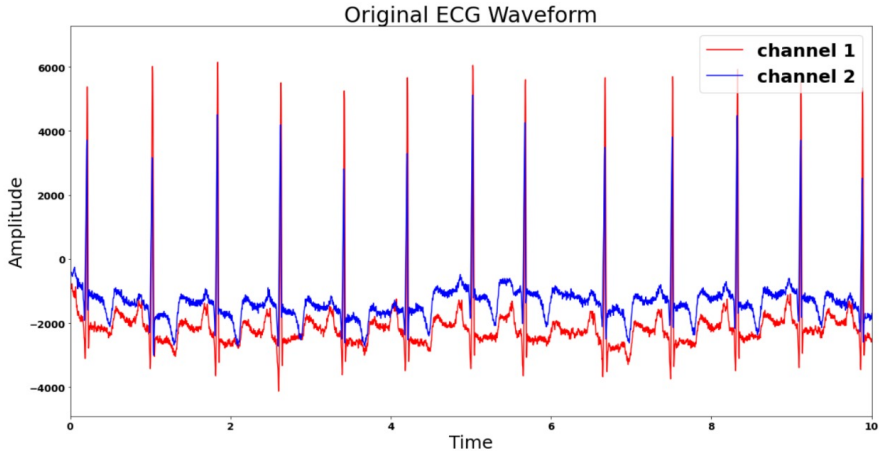

Fig. 2. Ten seconds of the original, unencrypted ECG Waveform

inverse. The square matrix is generated by padding zeroes, to the column matrix. However, if there are n rows in the sampled data of the column matrix, n-1 columns of zeroes would have to be added. In the patient record chosen for encryption, there are 1300000 rows of sampled data, meaning that 1299999 columns of zeroes would have to be padded. This operation is highly inefficient and makes computing the inverse of a 1300000 x 1300000 very expensive and time-consuming. To decrease the number of zeroes that are padded, it is proposed that the column matrix first be rearranged into an m x n matrix, with m rows and n columns. If r is the total number of rows in the column matrix, then m and n are chosen such that the product of m and n is r, and the difference between the smallest square matrix that can be made from the chosen dimensions and m x n is minimum. This would minimize the number of zeroes that have to be padded.

5 Reduction of matrix size

The program to compute the configuration of rows and columns in the matrix, that would lead to the least number of zeroes being padded is written using three user-defined functions. Initially, a function $factorList(n)$ computes all the factors of the total length of the column matrix. Next, the function $combination(n, factors)$, finds possible combinations such that the product of the ordered pair is equal to the number of rows in the column matrix. Then, the function optimize(row, col) is used to find out the side length of the smallest square matrix, that can be made from every ordered pair created. The number of zeroes that would have to be padded in each case is computed. For every set of ordered pairs created, the number of zeroes to be padded is calculated and the

pair which requires the least number of zeroes to be added is called the optimal pair. The column matrix is then reshaped in the optimal pair's dimensions.

5.1 Pseudo code

Function to find the factors of the sampled data length and to generate ordered pairs

factorList(n)

Algorithm 1 Find factors of sampled data length and generate ordered pairs

Input: n (number of rows in the column matrix), Array f_list, Array pairs
Output: Array with ordered pairs of factors

```
i ← 1
j ← 0
k ← 0
count ← 0
for i < n/2 do
  if n%i == 0 then
    f_list[count] ← i
    count ← count + 1
  end if
end for
index ← 0
factors_length ← length(f_list)
for j < factors_length do
  element ← factors[j]
  for k < factors_length do
    if element * factors[k] = n then
      if (factors[k], elements) not in pairs then
        pairs[index] ← (factors[k], elements)
        index ← count + 1
      end if
    end if
  end for
end for
```

Function to compute the smallest sized square matrix for a given row and column dimension, and thereby the number of zeros to be padded.

optimize(row,col)

Algorithm 2 To compute the smallest matrix size to minimize zero padding

Input: row,col, Array pairs

Output: A zero padded square matrix, with minimum dimensions
  $square\_size \leftarrow 0$
  $zero\_pad \leftarrow 0$
  $factors\_length \leftarrow length(f\_list)$
  if col >= row **then**
    $square\_size \leftarrow col$
  else
    $square\_size \leftarrow row$
  end if
  $zero\_pad \leftarrow (square\_size * square\_size) - (row * col)$
  $f\_list \leftarrow factorList(n)$
  $pairs \leftarrow combinations(n, f\_list)$
  $min\_zero \leftarrow None$
  $optimal\_pair \leftarrow pairs[0]$
  $i \leftarrow 0$
  for $i < factors\_length$ **do**
    $zeroes \leftarrow optimize(pairs[i][0], pairs[i][1])$
    if $min\_zero = None$ **then**
      min $\_zero \leftarrow zeroes$
    else
      if $min\_zero < zeroes$ **then**
        $min\_zero \leftarrow zeroes$
        $optimal\_pair \leftarrow pairs[i]$
      end if
    end if
  end for
  return $optimal\_pair$

6 Encryption and Decryption

The encryption of the ECG waveform is crucial during transmission from one healthcare facility to another. The process of encryption should be efficient, yet complex enough that it cannot be decrypted easily. The method proposed uses a symmetric key encryption technique. In this method, there is a single key, which must be kept secret from third parties. Symmetric key methods are fast, and are therefore used to encrypt bulk data. After the minimization of the size of the square matrix, a random key matrix of the same dimensions is generated. This key matrix is very large, and has random values in the form of doubles, making it very difficult to guess[9]. This key matrix is then multiplied with the generated square matrix, to obtain the encrypted matrix.

6.1 Pseudo code

Encryption

encrypt_wave(n,a)

Algorithm 3 To encrypt the square matrix using a random key

Input: matrix dimensions, wave file
Output: An encrypted matrix enc
$optimal\_pair \leftarrow optimize(row, col)$
Interchange the values of optimal_pair[0] and optimal_pair[1]
$a\_reshaped \leftarrow reshape(a, optimal\_pair[1], optimal\_pair[1]$
$flag \leftarrow 0$
if $optimal\_pair[1] > optimal\_pair[2]$ **then**
$sq\_wave \leftarrow horzcat(a\_reshaped, zeros(optimal\_pair[1], optimal\_pair[1] - optimal\_pair[2]))$
else
$flag \leftarrow 1$
$a\_reshaped \leftarrow transpose(a\_reshaped)$
$sq\_wave \leftarrow horzcat(a\_reshaped, zeros(optimal\_pair[2], optimal\_pair[2] - optimal\_pair[1]))$
end if
$key \leftarrow rand(size(sq\_wave))$
$enc \leftarrow sq\_wave * key$
return *enc*

The first ten seconds of the ECG .wav file, once encrypted, is as shown in Figure 4.

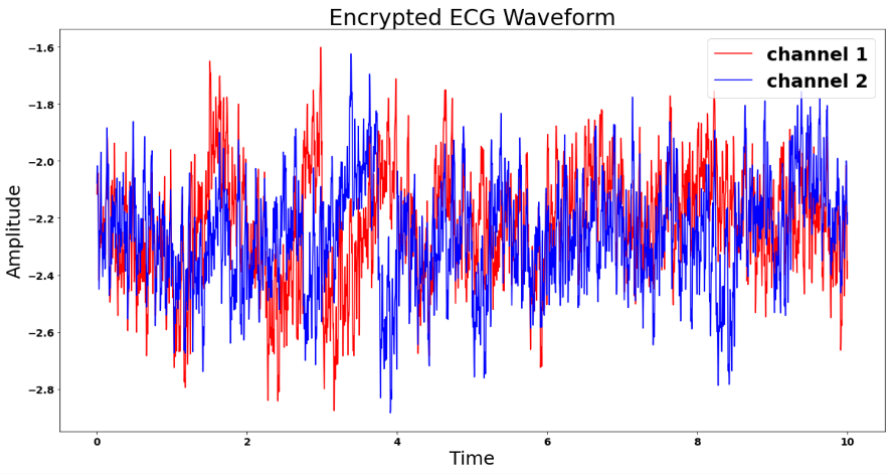

Fig. 3. Ten seconds of the encrypted ECG Waveform

Decryption The process of decryption involves computing every single value in the randomly generated key matrix. The inverse of the key matrix is computed, and then the padded zeroes are removed. The data is then reshaped back into its original two column format.

decrypt_ wave(enc,key,flag)

Function to compute the smallest sized square matrix for a given row and column dimension, and thereby the number of zeros to be padded.

decrypt_ wave(enc,key,flag)

Algorithm 4 To obtain the decrypted ECG waveform

Input: enc,key,flag (Indicates whether the matrix needs to be transposed or not)
Output: Array a
  $dec \leftarrow enc/key$
  Strip the zeroes in dec and store in a_stripped
  if $flag = 1$ **then**
    $a\_reshaped \leftarrow transpose(a\_stripped)$
  end if
  Reshape a_stripped such that it has only two columns and store it in a
  return a

7 Result and Analysis

The intention behind finding the smallest possible square matrix and then encrypting it drastically reduces the size of the matrix and the time taken to compute the inverse of the matrix during decryption. The efficiency of this method is analyzed by means of the average time taken to encrypt and decrypt in existing systems and in the proposed system. The accuracy of the system is also tested using evaluation metrics.

Algorithm Results The results of each algorithm tested on a ten-second sample are discussed in Table 1.

Table 1: Results of the Algorithms applied to the ECG Waveform

Algorithm	**Functionality**	**Result**
factorList(n)	To find factors of the total sampled data length and to generate suitable ordered pairs	On ten seconds of sampled data, there are **53** factors, and **26** possible ordered pairs.
optimize(row,col)	To find the most optimal of ordered pairs to minimize the padding of zeroes. The function identifies the minimum number of rows/columns in the square matrix. The number of zeroes is then computed. The condition which leads to the least number of zeroes being added is taken to be the optimal scenario.	For **7200** sampled values, in the ten seconds of the ECG waveform, the ordered pair **90 x 80** is found to the dimensions that lead to the least number of zeroes being padded. **900** zeroes have to be padded, rather than **51,832,800**, in the case of making a 7200 x 7200 matrix.
encrypt_wave(n,a)	Reshapes matrix according the optimal pair dimensions. The function also generates a random key, which is multiplied with the unencrypted matrix	The product matrix is of the same size as the reshaped matrix. For 10 seconds of the wave form, the encrypted matrix size is **90 x 90**.
decrypt_wave(enc,key,flag)	Computes the inverse of the key matrix, and multiples it with the encrypted matrix, to obtain the original waveform. A check is in place to transpose the matrix back to its original dimensions. The waveform is reshaped back into a column matrix.	The decrypted matrix has a size of **90 x 80**. The column matrix has **7200** rows, which is equivalent to the original waveform's size.

Efficiency The time taken for the encryption and decryption process is a key factor in the efficiency of a cryptographic technique. Ten-second samples from the ECG waveform are taken, and the average time taken for encryption and decrypting the ion is computed using the existing technique of padding multiple columns of zeroes to a single-column matrix, as well as the method of rearranging the columns such that a minimum number of zeroes is padded. The computation time in seconds for both methods is presented in Table 2.

Table 2. Comparison of Average Computation time of Different Methodologies

Process	**Padding to a column matrix (sec)**	**After reduction of matrix size (sec)**
Encryption	7.4301	0.0000469
Decryption	35.7486	0.0001865

There is a clear decrease in the time taken for encryption and decryption, due to the overall decrease in the matrix's size.

Evaluation Metrics In order to analyze the similarity between the original and decrypted waveforms, several metrics can be utilized. Histograms measure how robust the systems are against attacks. The better the system, the more the encrypted values should have a normal distribution [3]. The Mean Square Error can also be calculated between the original and decrypted data to evaluate any possible error. Other metrics include the Structural Similarity Index (SSIM), which compares the level of similarity between the original and the encrypted waveform.

Histogram In Figure 5, the distribution of the encrypted values is shown in the histogram. The plot shows a roughly normal distribution.

Mean Squared Error The Mean Squared Error (MSE) gives the average squared difference between the actual and the observed values. The MSE is computed with the encrypted and decrypted values, to measure the extent of error produced during decryption. Figure 6 shows a plot of the Mean Squared Error of the ECG waveform. The Mean Squared Error(MSE) is given by Equation 1.

$$MSE = \frac{1}{n}\sum_{k=1}^{n}(x_i - \bar{x}_i)^2 \tag{1}$$

where n is the number of samples taken, x_i is the original value, that is, the unencrypted amplitude value of the ECG waveform and $\overline{x_i}$ is the decrypted value of the waveform.

The calculated MSE between the decrypted and the encrypted values is found to be of the order 10-23, indicating a high level of accuracy.

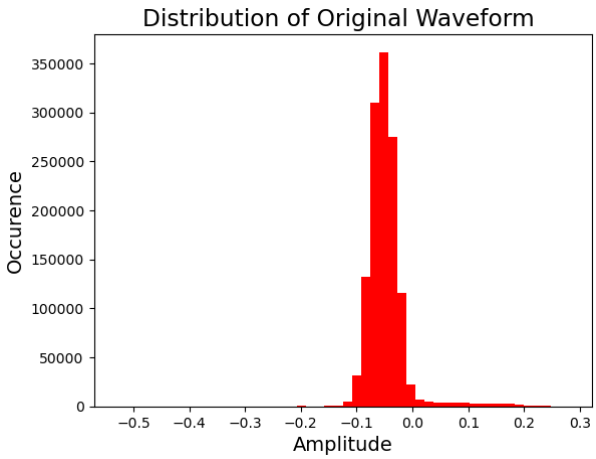

Fig. 4. Histogram of distribution of original waveform values

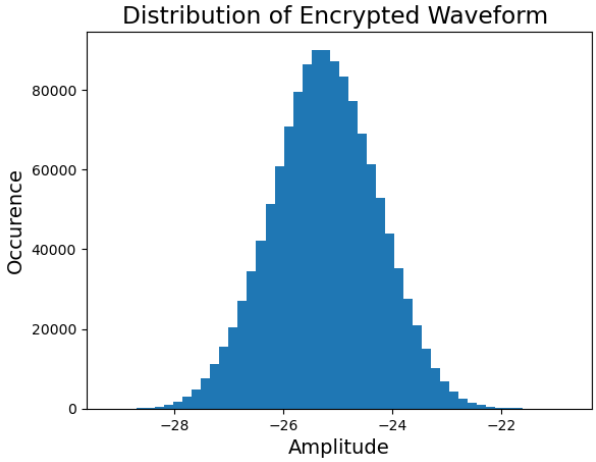

Fig. 5. Histogram of distribution of encrypted waveform values

Structural Similarity Index The Structural Similarity Index is used to analyze how similar the waveform's original and encrypted values are. An SSIM of 1 indicates that the wave forms are identical, whereas a value of 0 means that they are entirely different. The SSIM is given by the expression:

$$SSIM = \frac{\left(2\overline{XY} + C_1\right)\left(2d_{xy} + C_2\right)}{\left(\overline{X}^2 + \overline{Y}^2 + C_1\right) + \left(d_x^2 + d_y^2 + C_2\right)}$$

(2)

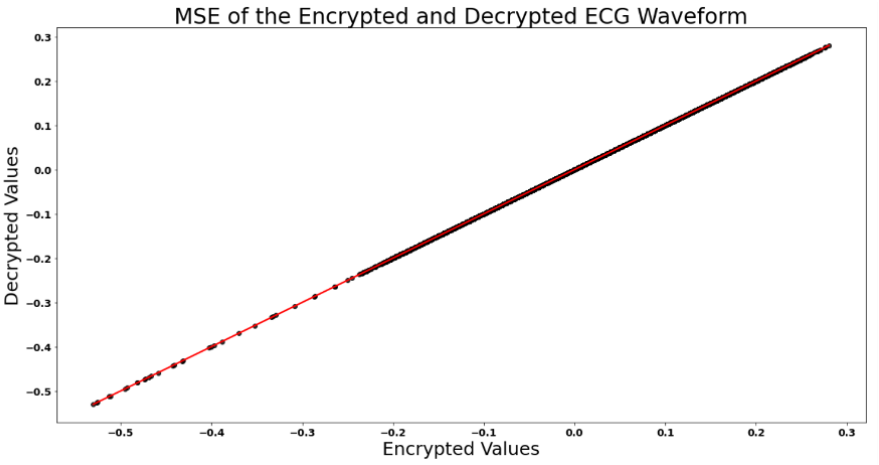

Fig. 6. Plot for Mean Squared Error of encrypted and decrypted wave forms

$\overline{X}$ and $\overline{Y}$ are the mean values of signal x and y. d_x and d_y are the variances of signal x and y respectively. C_1 and C_2 are some small values. d_{xy} is the cross-covariance between the signals x and y.

The values taken in the key matrix are between 0 and 1. The SSIM of the original and encrypted ECG wave forms is found to be 0.6.

Observation It can be seen that the margin of error is very slim, given that the key matrix is inverted for decryption. The probability of cracking the key is also negligible, given the very large size of the key matrix, and the fact that values in the key matrix are doubles. However, since the values lie between zero and one, the structural similarity is relatively high.

8 Conclusion

In this paper, a method has been proposed to make encryption and decryption using matrices less time-consuming. It is also a solution with very high levels of accuracy. This method can also be extended for other encryption techniques that use matrices for encryption, where there is a necessity for square matrices. The padding of zeroes to make the matrix square is largely minimized. The random key matrix generated is also of a large order, even after minimization of the zeroes padding, and the random values within it are doubles, making it secure, and thereby difficult to decipher. Additional testing on the efficacy of the solution can be done by mathematically modeling the QRS waveform[10] and the encrypted waveform. The proposed method facilitates the secure transmission of

ECG wave forms, without compromising security. It can effectively encrypt and decrypt .wav files of longer durations, without data loss.

References

1. Addison, P.S.: Wavelet transforms and the ecg: a review. Physiological Measurement **26**(5), R155 (aug 2005). https://doi.org/10.1088/0967-3334/26/5/R01, https://dx.doi.org/10.1088/0967-3334/26/5/R01
2. Al-laham, M.M.: A method for encrypting and decryptingwave files. International Journal of Network Security & Its Applications (IJNSA) Vol **10** (2018)
3. Algarni, A.D., Soliman, N.F., Abdallah, H.A., El-Samie, F.E.A.: Encryption of ecg signals for telemedicine applications. Multim. Tools Appl. **80**, 10679–10703 (2021)
4. Bai, T., Lin, J., Li, G., Wang, H., Ran, P., Li, Z., Li, D., Pang, Y., Wu, W., Jeon, G.: A lightweight method of data encryption in bans using electrocardiogram signal. Future Generation Computer Systems **92**, 800–811 (2019)
5. Hameed, M.E., Ibrahim, M.M., Abd Manap, N., Attiah, M.L.: Comparative study of several operation modes of aes algorithm for encryption ecg biomedical signal. International Journal of Electrical and Computer Engineering **9**(6), 4850 (2019)
6. Kaplan Berkaya, S., Uysal, A.K., Sora Gunal, E., Ergin, S., Gunal, S., Gulmezoglu, M.B.: A survey on ecg analysis. Biomedical Signal Processing and Control **43**, 216–235 (2018). https://doi.org/https://doi.org/10.1016/j.bspc.2018.03.003, https://www.sciencedirect.com/science/article/pii/S1746809418300636
7. Liu, T.Y., Lin, K.J., Wu, H.C.: Ecg data encryption then compression using singular value decomposition. IEEE Journal of Biomedical and Health Informatics **22**(3), 707–713 (2018). https://doi.org/10.1109/JBHI.2017.2698498
8. Moody, G.B., Mark, R.G.: The impact of the mit-bih arrhythmia database. IEEE Engineering in Medicine and Biology Magazine **20**(3), 45–50 (2001)
9. Nadir, J., Ein, A.A., Alqadi, Z.: A technique to encrypt-decrypt stereo wave file. International Journal of Computer and Information Technology **5**(5), 465–470 (2016)
10. Sornmo, L., Borjesson, P.O., Nygards, M.E., Pahlm, O.: A method for evaluation of qrs shape features using a mathematical model for the ecg. IEEE Transactions on Biomedical Engineering (10), 713–717 (1981)
11. Sufi, F., Han, F., Khalil, I., Hu, J.: A chaos-based encryption technique to protect ecg packets for time critical telecardiology applications. Security and Communication Networks **4**(5), 515–524 (2011)
12. Upadhyay, R.R.: Study of encryption and decryption of wave file in image formats. arXiv preprint arXiv:1307.6711 (2013)
13. Zhang, Z., Wang, H., Vasilakos, A.V., Fang, H.: Ecg-cryptography and authentication in body area networks. IEEE Transactions on Information Technology in Biomedicine **16**(6), 1070–1078 (2012)