Received March 11, 2021, accepted April 12, 2021, date of publication April 20, 2021, date of current version April 27, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3074254

Reversible Data Hiding in Encrypted DICOM Images Using Cyclic Binary Golay (23, 12) Code

MARIUSZ DZWONKOWSKI[1,2] AND ROMAN RYKACZEWSKI[2]
[1]Department of Radiology Informatics and Statistics, Faculty of Health Sciences, Medical University of Gdańsk, 80-210 Gdańsk, Poland
[2]Department of Teleinformation Networks, Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, 80-233 Gdańsk, Poland

Corresponding author: Mariusz Dzwonkowski (mard@gumed.edu.pl)

This work was supported by the Ministry Subsidy for Research, Gdańsk University of Technology.

ABSTRACT In this paper, a novel reversible data hiding method for encrypted images (RDHEI) is proposed. An efficient coding scheme based on cyclic binary Golay (23, 12) code is designed to embed additional data into the least significant bits (LSBs) of the encrypted image. The most significant bits (MSBs) are used to ensure the reversibility of the embedding process. The proposed scheme is lossless, and based on the receiver's privileges, allows recovery of marked data, original data and embedded data. Furthermore, the scheme can be used with any type of data, however it is best suited to 16-bit DICOM images of monochrome photometric interpretation. A modification to the standard DICOM network model was also introduced, to point out an example application of the proposed RDHEI method, i.e. an anonymized data storage outsourcing. A computer-based analysis has been carried out and simulation results are shown at the end of this paper.

INDEX TERMS DICOM, encrypted images, Golay code, lossless scheme, reversible data hiding, medical data.

I. INTRODUCTION

The Digital Image and Communication On Medicine (DICOM) standard was developed in order to facilitate safe and reliable transmissions and communications of medical imaging information. The DICOM standard relies on security techniques that include the Advanced Encryption Standard (AES) and the Triple Data Encryption Standard (3DES) algorithms [1]. Implementation of the encryption mechanisms introduces a problem with key distribution when sharing encrypted medical data with (not necessarily trusted) third parties outside the DICOM network. Reversible data hiding techniques for encrypted data allow additional data to be embedded reversibly without decryption. Such an approach not only allows for categorization of the data recipients, but also enables cloud-based data outsourcing for data management, authentication or other purposes where data embedding is performed on encrypted data.

In this paper, we present a novel RDHEI method for DICOM images. The concept of introducing data coding to enable data embedding for encrypted images, presented in [2], inspired us to research and implement our own scheme.

The associate editor coordinating the review of this manuscript and approving it for publication was Ramakrishnan Srinivasan.

The main purpose of this paper is to present a novel approach that features a very high embedding capacity (more than twice the embedding capacity shown in [2]) and quality of the marked data. The structure of the scheme is also much less complex, thus the proposed method is well suited for real-time applications in the cloud.

Because of the detailed comparison with other existing RDHEI methods, already shown in [2], we treated [2] as a reference for our comparison analysis. To the best of our knowledge, this is the first RDHEI method to use the cyclic binary Golay (23, 12) code for data embedding. The contribution of this paper is as follows.

- A cyclic binary Golay (23, 12) coding scheme, exploiting the properties of syndromes and error vectors to enable a high embedding capacity and quality of the marked data.
- A data recovery scheme, which enables a reconstruction of the original data (error-free) and recovery of the embedded data (before or after decryption).
- New data pre-processing operations i.e. a binary decomposition (dedicated for 16-bit DICOM images) and a block permutation, to rearrange the structure of a data string and thus facilitate the embedding process.

- A modified DICOM network with applied RDHEI to allow anonymized data storage outsourcing.

The structure of the paper is as follows. Related works are listed in Section 2. An example concept of a modified DICOM network with applied RDHEI mechanisms is shown in Section 3. Section 4 contains a brief description of the cyclic binary Golay (23, 12) code, its properties and coding rules. The proposed RDHEI method, with details on data processing, data embedding and data recovery is described in Section 5. Simulation results and analysis are presented in Section 6. The conclusions are discussed in Section 7.

II. RELATED WORK

RDH has a long, nearly 25-year history and initially focused on methods that hid data (typically an authentication data) in images, directly in the space domain. The techniques used, which can be called classical, rely on a small entropy value of the prediction error of pixel values due to their strong correlation. The additional data (after calculating the prediction errors) is placed in the image using various techniques [3]: histogram shifting [4]–[6], difference expansion [7]–[9], or pixel value ordering [10], [11].

Currently, due to the expanding range of applications, RDH methods are used to place various information in multimedia data i.e. information about authors, contractors, distributors, etc. In medicine, RDH methods are often used to enrich medical images with sections of the EPR (Electronic Patient Record), which is a digital version of a patient's paper chart (medical history, disease diagnosis, treatment methods, etc.) [12], [13]. Additionally, RDH methods are applied not only to images, but also to video signals [14], [15] and sound signals [16], [17]. A comprehensive review of the literature on RDH methods from 1997-2016 is presented in [3].

In this work, we deal with the particularly interesting and rapidly developing category of RDH methods in encrypted images (RDHEI) i.e. RDH performed on encrypted data (typically images). RDHEI methods can be divided into three main groups:

- reserving room before encryption (RRBE) [18],
- reserving room after encryption (RRAE) [19]–[22],
- methods utilizing homomorphic encryption [23]–[28].

RRBE methods, feature additional pre-processing of the original data (before encryption) to increase the embedding capacity and enable error-free recovery of the original data.

RRAE methods are more computationally efficient due to the lack of additional pre-processing of the original data, but feature a smaller embedding capacity and possible errors in the recovered data.

RDH methods that use homomorphic encryption like the additive homomorphicity of the Paillier cipher [23]–[26] feature a significant increase in data volume [23]. Methods for which there is no volume expansion [27], [28], feature a rather small embedding capacity of 0.2 bits per pixel (bpp) [28].

RDHEI methods can also be distinguished (from the user's perspective) as joint or separable. For joint RDHEI methods, a user with a decryption key obtains both the hidden data and the original data. For separable RDHEI methods, the recovery process of hidden data and original data requires two separate keys. If a user has a decryption key for the hidden data, they can obtain it directly from an encrypted image. If the user has only the image decryption key, they can obtain a marked image, i.e. an image containing hidden data, of a quality not much different from the original. If the user has both keys, they can obtain both the hidden data and the original image without any distortion.

The first separable RDHEI method was proposed in [29], which was later improved in many papers [2], [21], [30], [31]. In [2], a prediction scheme of MSB values (shown earlier in [32]) and the properties of the Hamming code (7, 4) were used to introduce a separable RDHEI-RRBE method with an embedding capacity of 0.452 bpp. The embedding process was performed on groups of 7 pixels (using their LSB values), derived from a group of applicable modifiable pixels, designated by the error prediction map. The reconstruction of the original image was based on the same groups of 7 pixels, but using their MSB values.

The method proposed in this paper was inspired by the work shown in [2]. Our method is also classified as a separable RDHEI-RRBE. We decided to use Golay (23, 12) code, which features a more efficient syndrome-source-coding than the Hamming (7, 4) code. Similarly to [2], we embed data in the LSB values of applicable pixels and reconstruct the original data based on the MSB values of these pixels, however, the applied embedding and reconstruction rules are different. The proposed embedding scheme exploits properties of Golay (23, 12) code to achieve an embedding capacity of almost 1 bpp and 2 bpp for 8-bit and 16-bit DICOM images, respectively. The proposed new pre-processing operations are also much less complex in computation, thus making the proposed RDHEI method more time-efficient and better suited for real-time cloud applications when compared with [2]. Additionally, for 8-bit images, the applied pre-processing data permutation provides robustness to the Ciphertext-Only Attack (exploiting inter-pixel redundancy when using a stream cipher), as thoroughly explained in [33].

III. MODIFIED DICOM NETWORK WITH APPLIED RDHEI

The rapidly growing importance of information emphasizes the issue of secure data storage and transfer in every modern hospital. Sharing medical data (raw or encrypted) between entrusted third parties cannot be done without first removing any PHI (Protected Health Information) to protect the subject's privacy, complying with HIPAA (Health Insurance Portability and Accountability Act) [34] rules, GDPR (General Data Protection Regulation) [35] rules and other data privacy regulations. An anonymization process needs to be performed to enable external use of medical data.

We propose a new DICOM network model, in which security applies not only to external connections (as is the case in commonly used solutions [36], [37]) but also to data storage.

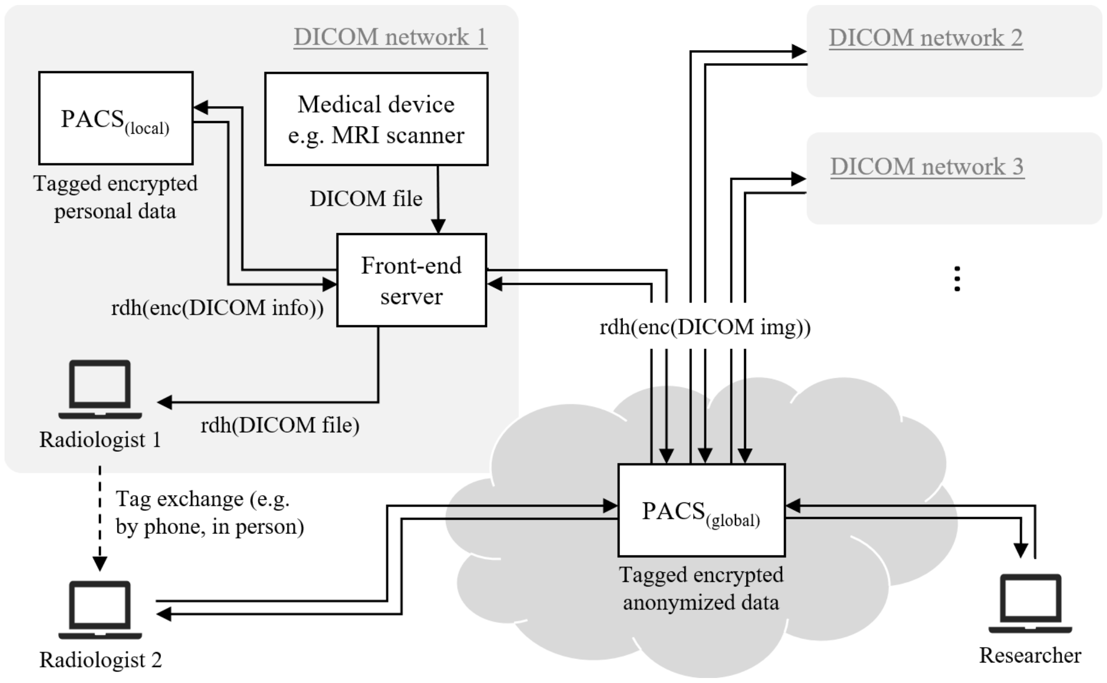

FIGURE 1. **A modified DICOM network with applied RDHEI.**

Additionally, by applying RDHEI mechanisms, it is possible to enable encrypted and anonymized data storage outsourcing along with data re-identification. It should be noted that the proposed modification of the DICOM network (Fig. 1) is not the only practical application of RDHEI in medicine, but one of many.

As shown in Fig. 1, the newly introduced elements and their functionality are as follows.

Front-end server is a management server whose tasks can be summarized as:

- receives DICOM files from any medical device inside a local DICOM network,
- separates the info part (entirely or only the PHI sections) from the image part of each received DICOM file,
- encrypts (with a stream cipher) the info part and the image part,
- tags (using the RDHEI method) the encrypted info part and the encrypted image part,
- sends the encrypted and tagged info part to the local PACS (Picture Archiving and Communication System) storage server,
- sends the encrypted and tagged image part to the global PACS storage server,
- forwards a tagged DICOM file to a radiologist inside the local DICOM network,
- reconstructs a DICOM file by combining the decrypted info part (downloaded from the local PACS server) with the decrypted image part (downloaded from the global PACS server) after matching their tag descriptions,
- controls external data access.

Although the DICOM standard defines security in several fields [1], [38], the use of the proposed front-end server eliminates the basic weakness of the standard DICOM network model, i.e. the possibility of accessing unsecured data on the storage server [39].

The global PACS storage server is a cloud-based repository of anonymized, tagged and encrypted DICOM images from different hospital networks. The repository can be accessed by any authorized entity i.e. radiologist, university researcher, etc. An authorized entity can search the repository using the tag information (which can be substantially detailed due to the proposed RDHEI method) and decrypt the selected data. They can even update the tag information in real-time (e.g. by embedding additional data without erasing the original tag). Because the global PACS server is meant to store a vast amount of medical data from different hospitals, a detailed data description via a tag (which does not consume storage space) is crucial for such a solution.

The local PACS storage server only stores the encrypted and tagged info part (personal information) of the DICOM files acquired inside hospital network.

The proposed network scenario not only provides researchers with easy access to anonymized DICOM images, but also benefits radiologists who want to consult their diagnosis with colleges from different hospitals. By exchanging only the tag information, other radiologists can find the right anonymized DICOM image on the global PACS server.

IV. CYCLIC BINARY GOLAY (23, 12) CODE

Binary Golay (23, 12) code was first introduced by Golay [40] in 1949. It is a perfect linear code capable of correcting any combination of three or fewer random errors in a block of 23 elements. The notation (n, k) means that it is a block code, with block length $n = 23$ bits, message information length $k = 12$ bits and redundant information (parity check) length $n - k = 11$ bits.

To define a cyclic binary Golay (23, 12) code G let us consider an irreducible polynomial over GF(2):

$$g(x) = \prod_{i=0}^{10} (x - \alpha^{2^i}) = x^{11} + x^9 + x^7 + x^6 + x^5 + x + 1. \tag{1}$$

There exists an element $\alpha \in \mathrm{GF}(2^{11})$ such that $g(\alpha) = 0$. Hence, the elements of $\mathrm{GF}(2^{11})$ are defined by the following set:

$$\{a_0 + a_1\alpha + a_2\alpha^2 + \cdots + a_{10}\alpha^{10}\}, \quad \text{where } a_0, a_1, \ldots, a_{10} \in \mathrm{GF}(2). \tag{2}$$

Element α is a primitive 23rd root of unity in $\mathrm{GF}(2^{11})$. This shows that $g(x)$ generates a cyclic Bose-Chaudhuri-Hocquenghem (BCH) block code of length 23, the so called Golay code [41].

Every valid codeword of Golay code G must be a multiple of generator polynomial $g(x)$. To represent a codeword, one can use a polynomial form or a vector form:

$$c(x) = \sum_{i=0}^{22} c_i x^i \quad \text{or } \mathbf{c} = [c_0, c_1, \ldots, c_{22}] \tag{3}$$

where $c_i \in \mathrm{GF}(2)$. For each codeword of Golay code G, there exists a cyclic invariance, meaning that a cyclic shift of the codeword by any number of bits will yield a valid Golay codeword. Additionally, by inverting the codeword, one can also obtain a valid codeword.

In order to encode a message into a codeword, using the matrix representation, one must first define generator matrix $\mathbf{G}$, obtained by cyclically shifting (to the left) generator polynomial $g(x)$ (written as a vector) inside a zero matrix of size $k \times n$ (12×23 for Golay code G):

$$\mathbf{G} = \begin{bmatrix} 1 \cdot g(x) \\ x \cdot g(x) \\ x^2 \cdot g(x) \\ \cdots \\ x^{k-1} \cdot g(x) \end{bmatrix}_{(12\times 23)}. \tag{4}$$

By using linear operations on the rows of matrix $\mathbf{G}$, one can convert $\mathbf{G}$ to a standard (canonical) form, i.e. $\mathbf{G} = [\mathbf{I}_{(12\times12)}|\mathbf{P}_{(12\times11)}]_{(12\times23)}$, represented by identity matrix $\mathbf{I}$ and matrix $\mathbf{P}$:

$$\mathbf{P} = \begin{bmatrix} 1&0&1&0&1&1&1&0&0&0&1 \\ 1&1&1&1&1&0&0&1&0&0&1 \\ 1&1&0&1&0&0&1&0&1&0&1 \\ 1&1&0&0&0&1&1&1&0&1&1 \\ 1&1&0&0&1&1&0&1&1&0&0 \\ 0&1&1&0&0&1&1&0&1&1&0 \\ 0&0&1&1&0&0&1&1&0&1&1 \\ 1&0&1&1&0&1&1&1&1&0&0 \\ 0&1&0&1&1&0&1&1&1&1&0 \\ 0&0&1&0&1&1&0&1&1&1&1 \\ 1&0&1&1&1&0&0&0&1&1&0 \\ 0&1&0&1&1&1&0&0&0&1&1 \end{bmatrix}. \tag{5}$$

The standard form of generator matrix $\mathbf{G}$ generates systematic codewords, meaning that message information k will be listed first in the codeword and will be separable from the redundant information contained in the remaining $n - k$ bits. To encode vector message $\mathbf{m}$, one must multiply it (mod 2) by generator matrix $\mathbf{G}$:

$$\mathbf{c}_{(1\times23)} = \mathbf{m}_{(1\times12)} \cdot \mathbf{G}_{(12\times23)}. \tag{6}$$

When decoding received word $\mathbf{r}$, a decoder must first determine if the word is correct or corrupted with errors. If received word $\mathbf{r}$ is correct, the decoder only needs to get the first k bits from $\mathbf{r}$ to obtain message $\mathbf{m}$. However, if received word $\mathbf{r}$ is corrupted, then the decoder needs to find and correct potential errors. There are many decoding algorithms for Golay (23, 12) code. For the purpose of the proposed RDHEI method, we present a decoding based on the standard table for Golay code G.

Decoding received word $\mathbf{r}$ comes down to calculating syndrome $\mathbf{s}$, which indicates error vector $\mathbf{e}$. Having the error vector, one can correct received word $\mathbf{r}$ to codeword $\mathbf{c}$ by adding (bitwise mod 2) vector $\mathbf{r}$ to vector $\mathbf{e}$: $\mathbf{r} \oplus \mathbf{e} = \mathbf{c}$. In order to calculate syndrome $\mathbf{s}$, using matrix representation, it is necessary to introduce parity check matrix $\mathbf{H}$:

$$\mathbf{H} = \left[\mathbf{P}^{\mathbf{T}}_{(11\times12)} | \mathbf{I}_{(11\times11)}\right]_{(11\times23)} \tag{7}$$

where $\mathbf{P}^{\mathrm{T}}$ is a transposition of matrix $\mathbf{P}$ from the standard form of generator matrix $\mathbf{G}$. A syndrome of received word $\mathbf{r}$ is calculated by multiplying (mod 2) vector $\mathbf{r}$ by the transposition of matrix $\mathbf{H}$:

$$\mathbf{s}_{(1\times11)} = \mathbf{r}_{(1\times23)} \cdot \mathbf{H}^{\mathbf{T}}_{(23\times11)}. \tag{8}$$

The calculated syndrome $\mathbf{s}$ indicates error vector $\mathbf{e}$. A standard table for Golay code G (which is based on matrix $\mathbf{H}$) is created to assign all possible syndromes (2^{11}) to all possible, correctable error vectors (23 combinations for 1 error, 22·23 for 2 errors and 21·22·23 for 3 errors). A section of the standard table for Golay code G is shown in Table 1.

Syndrome $\mathbf{s} = 0$ points at error vector $\mathbf{e} = 0$, which means that received word $\mathbf{r}$ is codeword $\mathbf{c}$, and there were no errors.

TABLE 1. Section of the standard table for Golay code *G*.

Errors (vectors **e** of length 23 bits)	Syndromes (vectors **s** of length 11 bits)
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1	0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0	0 0 0 0 0 0 0 0 0 1 0
…	…
0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0	1 1 1 1 1 0 0 1 0 1 1
…	…
0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0	1 1 1 1 0 1 1 1 0 0 0
…	…

V. PROPOSED SCHEME

Based on the properties of the Golay coding (presented in Section 4) we introduce a novel, error-free RDHEI-RRBE method, which allows additional data to be embedded in encrypted DICOM images. The method operates on raw DICOM files (without a lossy/lossless compression) to provide a processing compatibility, as many DICOM viewers can only handle uncompressed raw data [42].

The method retains the separable property, i.e. it provides 3 different scenarios on the receiver's side (Fig. 2).

The proposed scheme can be summarized by 3 main stages: data processing (sender's side), data embedding (data hider's side) and data recovery (receiver's side). Each stage is discussed in detail in the following Subsections 5.1.-5.3.

A. DATA PROCESSING

In the first stage, called data processing, the input data must first be properly prepared (pre-processed) before encryption. When the input data is a 16-bit DICOM image, such an image needs to be binary decomposed into two 8-bit images. All even bits will form one image and all odd bits will form the other, as shown in Fig. 3. Thus, each 8-bit image will possess MSB values derived from two sets of MSBs of the 16-bit DICOM image. For DICOM images with a monochrome photometric interpretation, those two sets of MSBs almost always equal 0, because pixel values are commonly represented by only 12 bits.

When the input data is a DICOM image with a different photometric interpretation i.e. 8-bit RGB or Palette Color, the binary decomposition step is omitted.

The next step, called block permutation, is performed on the 8-bit data. The data is divided into blocks of 23 pixels (or 8-bit values in general, in the case that the input data is not an image). The MSB values of all pixels are checked in each block (the last block is omitted if it contains fewer than 23 pixels). Each block that contains all 23 MSB values equal to 0 or 1 are moved to the beginning of the data string and these blocks will be used for data embedding (Fig. 4).

Information about the number of rearranged blocks and their initial position is saved as a permutation key. The permutation key is a binary vector of size $1 \times \lfloor M \cdot N/23 \rfloor$ (where $M \times N$ is the size of the input image) and its elements show whether the block is applicable for data embedding (value 1) or not (value 0). It should be noted, that in the case of a monochrome 16-bit DICOM image, each of the two 8-bit images (obtained after binary decomposition) will yield MSB values equal to 0 for each pixel in every block, thus there will be no need for block permutation and every block will be used for data embedding.

The next step after data pre-processing is data encryption. To enable the RDHEI approach, a stream cipher is used for a bitwise data encryption. To encrypt an n-bit data string, one must generate n pseudo-random bits (based on the encryption key) and perform a bitwise XOR encryption. An exception must be made for data representing pixels from rearranged blocks (i.e. blocks that contain all 23 MSB values equal to 0 or 1). In this case, XOR encryption is not performed on two LSBs of each pixel from such a block (Fig. 5). It should be noted that in the case of a monochrome 16-bit DICOM image, each of the two 8-bit images (obtained after binary decomposition) will be encrypted entirely in that manner.

By using the unencrypted LSBs (which are similar to random bits and reflect almost no meaningful content), one will be able to obtain embedded data from either the encrypted image or the marked decrypted image (this is thoroughly explained in Subsection 5.3).

B. DATA EMBEDDING

In the second stage called data embedding, additional data is only added to partially encrypted blocks (Fig. 5) of the encrypted data from the first stage. A permutation key (defined during the first stage) is needed to know the exact number of partially encrypted blocks. However, in case of a monochrome 16-bit DICOM image, all received blocks are partially encrypted and thus applicable for data embedding.

In each partially encrypted block, one can embed up to 22 bits of additional data. An embedding key, introduced at this stage, retains information about the number of partially encrypted blocks and determines the order in which the partially encrypted blocks are selected for embedding additional data.

Before embedding additional data, it is necessary to embed the permutation key (provided that the block permutation step was conducted). The permutation key size is always known (based on image size $\boldsymbol{M} \times \boldsymbol{N}$), thus there is no need to use a binary separator for the additional data that is embedded next. Additionally, partially encrypted blocks with an embedded permutation key are not affected by the embedding key, i.e. their order is fixed and their position is always at the beginning of the data string (Fig. 6).

The embedding process can be summarized as follows.

First, treat the last two rows of each partially encrypted block as two 23-bit words $\mathbf{r}_1$ and $\mathbf{r}_2$ of Golay code G and calculate the syndromes of these words according to equation (8). Record the calculated $\mathbf{s}_1$ and $\mathbf{s}_2$ syndromes in the MSB positions of 22 pixels in the same partially encrypted block (Fig. 7). The MSB of the first pixel must be left unchanged.

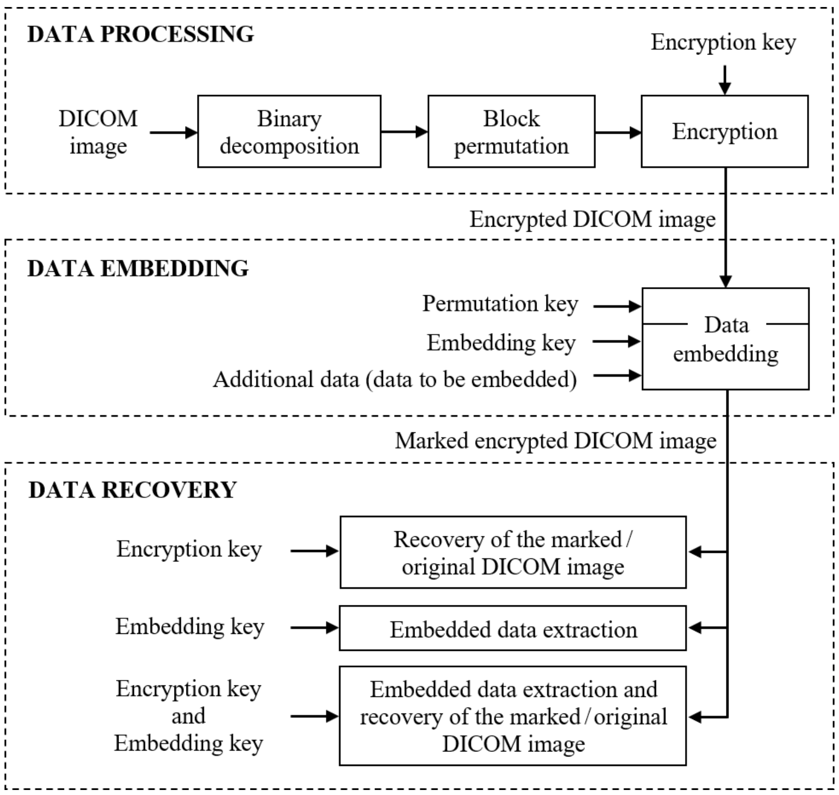

FIGURE 2. Overview of the proposed RDHEI scheme for DICOM images.

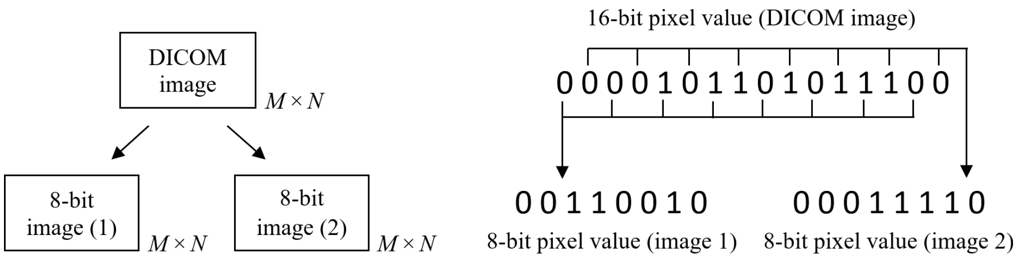

FIGURE 3. Binary decomposition of a 16-bit DICOM image.

Divide a 22-bit section of additional data (data to be embedded) into two 11-bit vectors called $\mathbf{a}_1$ and $\mathbf{a}_2$. Calculate two 11-bit vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ by adding (bitwise mod 2) corresponding vectors $\mathbf{s}_i$ and $\mathbf{a}_i$:

$$\mathbf{x}_1 = \mathbf{s}_1 \oplus \mathbf{a}_1 \quad \mathbf{x}_2 = \mathbf{s}_2 \oplus \mathbf{a}_2. \tag{9}$$

Based on standard table for Golay code G (Table 1), find error vectors $\mathbf{e}_1$ and $\mathbf{e}_2$ for syndromes $\mathbf{x}_1$ and $\mathbf{x}_2$. Calculate two 23-bit vectors $\mathbf{m_1}$ and $\mathbf{m_2}$ by adding (bitwise mod 2) corresponding vectors $\mathbf{r}_i$ and $\mathbf{e}_i$:

$$\mathbf{m}_1 = \mathbf{r}_1 \oplus \mathbf{e}_1 \quad \mathbf{m}_2 = \mathbf{r}_2 \oplus \mathbf{e}_2. \tag{10}$$

Replace vectors $\mathbf{r}_1$ and $\mathbf{r}_2$ with vectors $\mathbf{m}_1$ and $\mathbf{m}_2$ in the partially encrypted block (a maximum of 3 bit flips for each row). The additional data $[\mathbf{a}_1, \mathbf{a}_2]$ is now successfully embedded into one partially encrypted block. Repeat the process for all remaining partially encrypted blocks by embedding other 22-bit sections of additional data.

C. DATA RECOVERY

The last stage called data recovery allows for three different scenarios as shown in Fig. 2. Each scenario is based on the privileges set for the receiver of the marked encrypted data.

In the first scenario, the receiver is only equipped with an encryption key. They will be able to recover a marked image (distorted 8-bit data) or an original image (undistorted 8-bit data). If the recovered data is an 8-bit decomposition of the 16-bit DICOM image, the data must be reassembled to create a 16-bit DICOM image (Fig. 3).

The received marked encrypted data is represented by a group of entirely encrypted binary blocks and partially encrypted binary blocks (placed at the beginning of the data

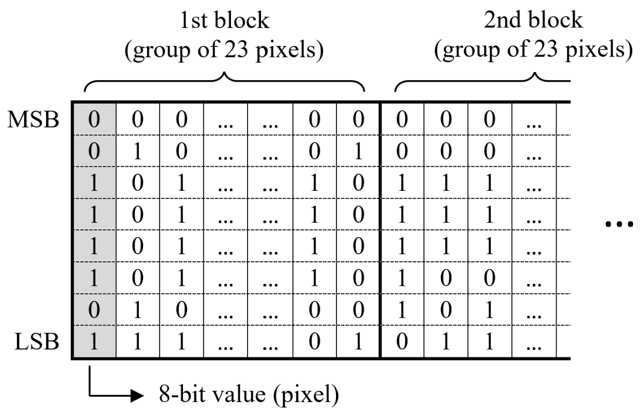

FIGURE 4. Input data decomposition into blocks of 23 pixels (8-bit values).

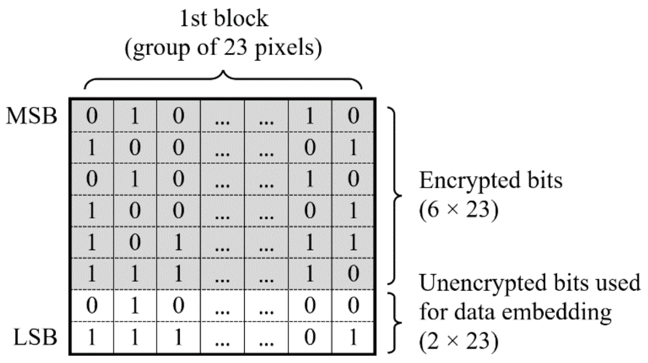

FIGURE 5. A partially encrypted binary block used for data embedding.

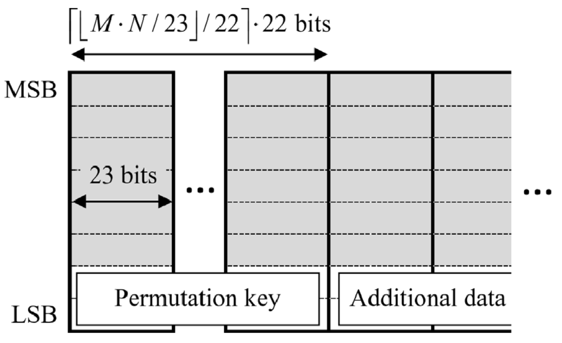

FIGURE 6. Positioning of embedded permutation key and additional data in the data string.

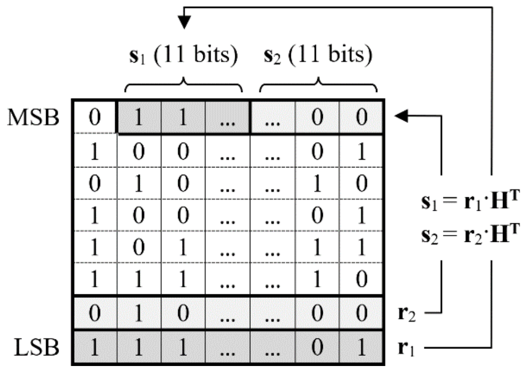

FIGURE 7. Placement of syndromes calculated from unencrypted bits of partially encrypted block.

string). Both binary blocks are decrypted using a bitwise XOR decryption. However, the partially encrypted binary blocks must be handled with exceptions as follows.

For each partially encrypted block, first calculate 11-bit vectors $\mathbf{a}_1$ and $\mathbf{a}_2$ (syndromes of vectors $\mathbf{m}_1$ and $\mathbf{m}_2$, Fig. 8):

$$\mathbf{a}_1 = \mathbf{m}_1 \cdot \mathbf{H}^{\mathbf{T}} \quad \mathbf{a}_2 = \mathbf{m}_2 \cdot \mathbf{H}^{\mathbf{T}}. \tag{11}$$

Next, calculate 11-bit vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ as shown in (9). Using the standard table of Golay code G, find error vectors $\mathbf{e}_1$ and $\mathbf{e}_2$ for syndromes $\mathbf{x}_1$ and $\mathbf{x}_2$. Calculate original vectors $\mathbf{r}_1$ and $\mathbf{r}_2$ by adding (bitwise mod 2) corresponding vectors $\mathbf{m}_i$ and $\mathbf{e}_i$:

$$\mathbf{r}_1 = \mathbf{m}_1 \oplus \mathbf{e}_1 \quad \mathbf{r}_2 = \mathbf{m}_2 \oplus \mathbf{e}_2. \tag{12}$$

Replace vectors $\mathbf{m}_1$ and $\mathbf{m}_2$ with vectors $\mathbf{r}_1$ and $\mathbf{r}_2$ in the partially encrypted block (Fig. 8). Perform a bitwise XOR decryption on the partially encrypted block but leave the two rows of LSBs, i.e. vectors $\mathbf{r}_1$ and $\mathbf{r}_2$, unchanged.

The decrypted MSB value of the first pixel in the block stores information about the MSB values of the next 22 pixels in that block. If the decrypted MSB value of the first pixel equals 1 (or 0), set the next 22 MSB values to 1 (or 0).

By performing the above steps for all partially encrypted blocks, decrypting all entirely encrypted blocks and then restoring the original order of all blocks of the data string with the permutation key, one will be able to successfully recover the original image.

It is important to note, that even though the receiver is not equipped with an embedding key, they will be able to recover the embedded permutation key, which is always embedded first, and its size is always known.

The recovery process of the marked image is the same as for the original, except for the step of calculating syndromes $\mathbf{a}_1$ and $\mathbf{a}_2$ to restore vectors $\mathbf{r}_1$ and $\mathbf{r}_2$ for the partially encrypted blocks, which is omitted. However, the recovery of the embedded permutation key is still necessary. For every partially encrypted block, one must just perform a bitwise XOR decryption, leaving two rows of LSBs unchanged, and then restore the MSB values (based on the decrypted MSB value of the first pixel in each block).

The marked image differs from the original due to two (possibly) modified LSB values of pixels from the partially encrypted blocks. After recovering the marked image, one can still extract the embedded data (provided the embedding key is known).

In the second scenario, the receiver is only equipped with an embedding key. They will be able to recover only the embedded data from the received marked encrypted image. The receiver must calculate syndromes $\mathbf{a}_1$ and $\mathbf{a}_2$ of two vectors $\mathbf{m}_1$ and $\mathbf{m}_2$ from every partially encrypted block, as shown in (11). The acquired vector $[\mathbf{a}_1, \mathbf{a}_2]$ is a 22-bit section of the embedded data. It should be noted that the embedding key is not needed for this operation. However,

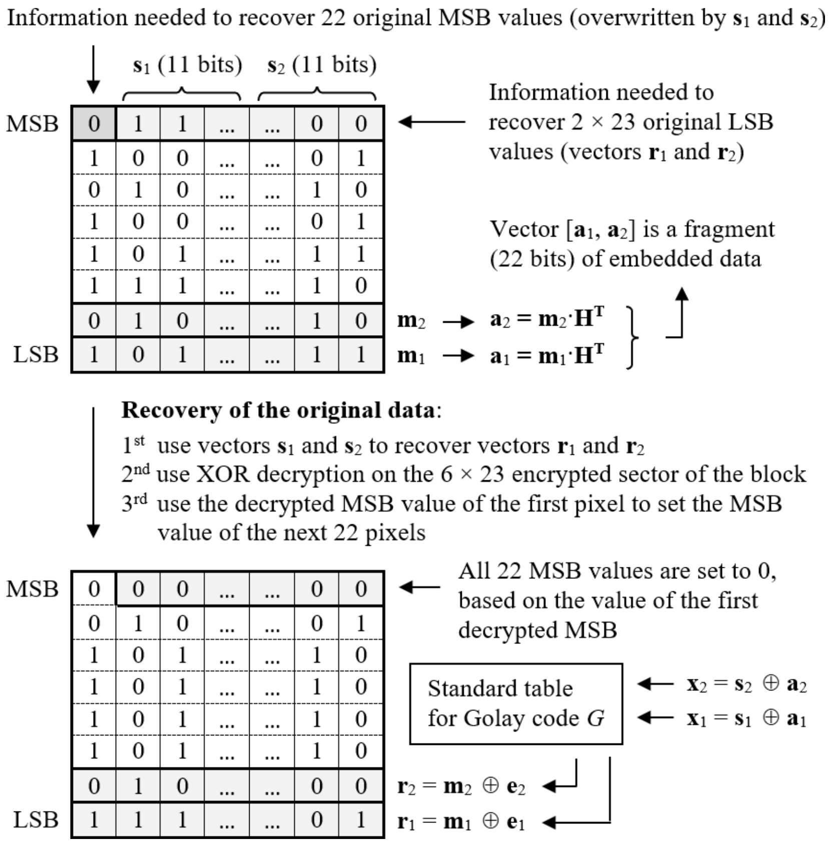
Information needed to recover 22 original MSB values (overwritten by s1 and s2)
s1 (11 bits)
s2 (11 bits)
MSB
LSB
Information needed to recover 2 × 23 original LSB values (vectors r1 and r2)
Vector [a1, a2] is a fragment (22 bits) of embedded data
m2 → a2 = m2·H^T
m1 → a1 = m1·H^T
Recovery of the original data:
1st use vectors s1 and s2 to recover vectors r1 and r2
2nd use XOR decryption on the 6 × 23 encrypted sector of the block
3rd use the decrypted MSB value of the first pixel to set the MSB value of the next 22 pixels
All 22 MSB values are set to 0, based on the value of the first decrypted MSB
Standard table for Golay code G
x2 = s2 ⊕ a2
x1 = s1 ⊕ a1
r2 = m2 ⊕ e2
r1 = m1 ⊕ e1

FIGURE 8. Recovery of the original data, performed on a partially encrypted block.

the embedding key is necessary to combine (in the correct order) all the extracted sections, i.e. vectors [$\mathbf{a}_1$, $\mathbf{a}_2$] into one message. The rearrangement process (via the embedding key) is not performed on vectors [$\mathbf{a}_1$, $\mathbf{a}_2$], which store information about the permutation key. These vectors are always sequentially extracted (in ascending order) from the partially encrypted blocks located at the beginning of the data string. The length of the permutation key is always known, thus the receiver knows where the additional data embedding begins.

In the third scenario, the receiver is equipped with both keys, i.e. the encryption key and the embedding key. They will be able to extract the embedded data and recover the original image, as described in the first scenario.

VI. SIMULATION RESULTS

A computer-based simulation was used to scrutinize the proposed scheme. Tests were performed on different types of DICOM files such as: Computed Tomography (CT), Magnetic Resonance (MR), Computed Radiography (CR), X-Ray Angiography and X-Ray Radiofluoroscopy, acquired from [43], [44]. All tests yielded similar results.

To maintain a clear presentation, the beginning of this section concerns tests for an example 16-bit CR DICOM image of monochrome photometric interpretation and of size 440 × 440 px. The image was decomposed into two 8-bit images. Each 8-bit image was encrypted and then marked with embedded data. The amount of embedded data for each encrypted 8-bit image was equal to 22.6 kB (8417 partially encrypted blocks × 22 bits), which is a bpp of 0.95648 and thus a bpp of 1.91295 for the 16-bit DICOM image.

The decryption process was performed for two cases. In the first case, the decryption was performed without removing the embedded data, which yielded marked 8-bit images with a PSNR above 52 dB, and a marked 16-bit reconstructed DICOM image with a PSNR above 88 dB. In the second case, the decryption was performed along with the recovery of the original data, and therefore the decrypted images were error-free.

The results of the encryption and decryption processes for the 16-bit DICOM image (decomposed into two 8-bit images) are shown in Fig. 9. Histograms of 8-bit images (original and encrypted from Fig. 9) are shown in Fig. 10. According to Fig. 10, the pixel values of the encrypted images are governed by a uniform distribution. Additionally, both histograms were measured by a Pearson's chi-squared test yielding p-values of 0.194 (b1) and 0.416 (b2) (more information about p-values is included in Subsection 6.1).

A. ANALYSIS OF RANDOMNESS FOR ENCRYPTED DICOM IMAGES

Chi-squared tests of randomness based on the diehard package [45] were implemented to analyse the randomness of

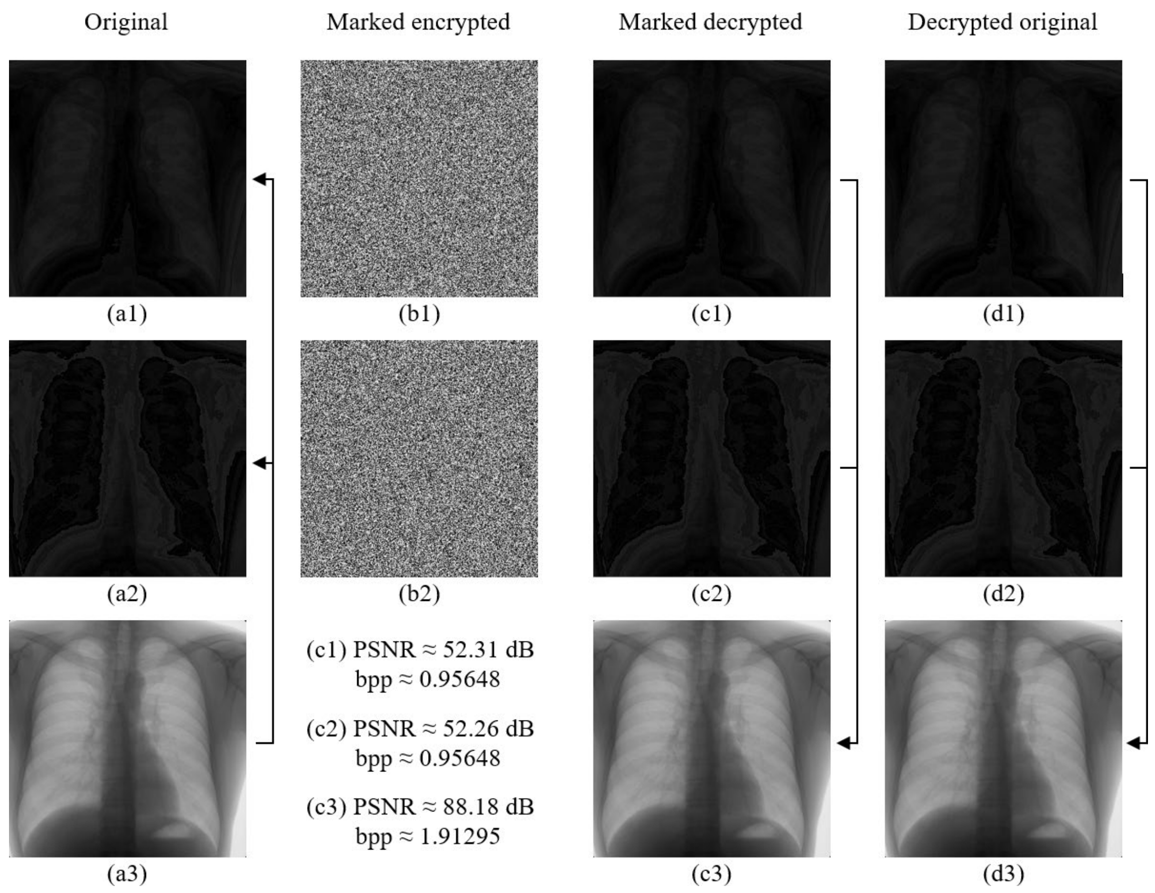
Original
Marked encrypted
Marked decrypted
Decrypted original
(a1)
(b1)
(c1)
(d1)
(a2)
(b2)
(c2)
(d2)
(c1) PSNR ≈ 52.31 dB bpp ≈ 0.95648
(c2) PSNR ≈ 52.26 dB bpp ≈ 0.95648
(c3) PSNR ≈ 88.18 dB bpp ≈ 1.91295
(a3)
(c3)
(d3)

FIGURE 9. Encryption and decryption performed on a 16-bit DICOM image: (a1-2) – original 8-bit images obtained from the original 16-bit DICOM image (a3), (b1-2) – encrypted 8-bit images with embedded data, (c1-2) – decrypted 8-bit images with embedded data, (c3) – reconstructed 16-bit DICOM image with embedded data, (d1-2) – decrypted 8-bit images without any errors, (d3) – reconstructed 16-bit DICOM image without any errors. Image source: [44].

TABLE 2. P-values for different randomness tests from the diehard package.

	Birthday spacings	Binary rank	Parking lot	The 3Dsphere	Up-down runs	Craps	OPERM5	Minimum distance	Overlapping sums
p-values	0.725	0.592	0.891	0.036	0.711	0.129	0.998	0.516	0.634

encrypted 16-bit DICOM images with embedded data. The primary factor that gives us information about the effectiveness of the obtained randomness in the encrypted data is a parameter called the p-value. Its value ranges from 0 to 1. If the p-value equals a strict 0 or 1 (by rounding to 6 decimal places), that means that the encrypted data fails a particular randomness test [45]. If the obtained p-value is very close to 0 or 1 (assuming a reasonable significance level, i.e. 5%) the test is passed. A value of 0.5 is the most desired outcome, but it is not a requirement to pass the test. The obtained results for selected diehard randomness tests are shown in Table 2.

We also analysed the randomness property of our algorithm by using a freeware application called Cryptool [46]. Cryptool offers 6 statistical tests: Frequency Test, Poker Test, Runs Test, Serial Test, and two additional tests embedded in Cryptool's battery test: Long-Run Test and Mono-Bit Test. The analysed encrypted data successfully passed, for a significance level of 5%, all the tests provided in Cryptool.

B. ANALYSIS OF MINIMUM PSNR FOR DICOM IMAGES

PSNR values were analysed for a decrypted 8-bit image with embedded data and a reconstructed 16-bit DICOM image with embedded data. Let us assume that the total number of pixels of the 16-bit DICOM image is a multiplication of 23 and let us assume that each block (group) of 23 pixels is used for data embedding. To find the minimum PSNR one must first determine the maximum mean squared error (MSE).

The most undesired outcome (maximum MSE) for an 8-bit image occurs when the embedded data is associated with

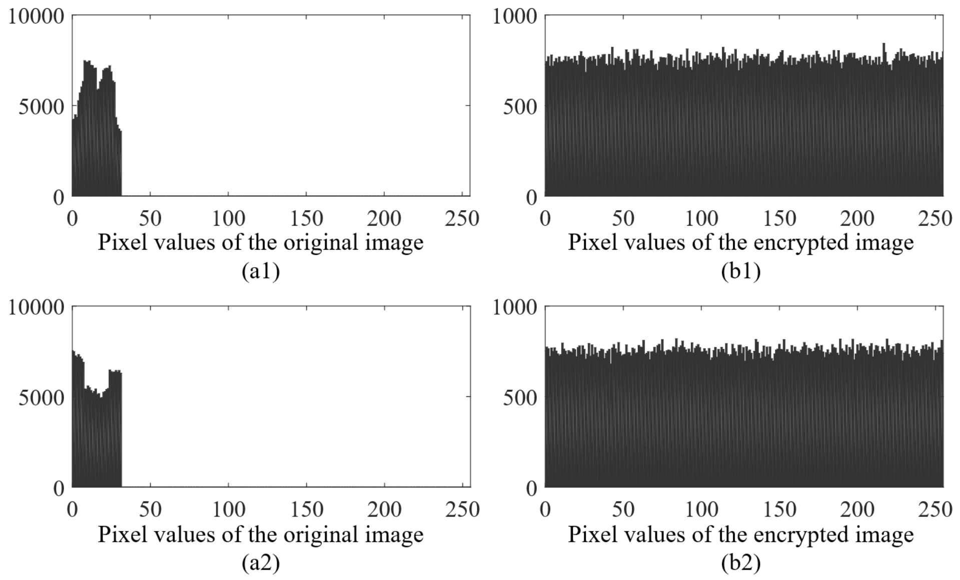

FIGURE 10. Histograms of original (a1-2) and encrypted with embedded data (b1-2) 8-bit images from Fig. 9.

the maximum number of errors, i.e. vectors **e** with three 1s. During the embedding process, two error vectors $\mathbf{e}_i$ are added (bitwise mod 2) to the two corresponding original vectors $\mathbf{r}_i$ in each block of 23 pixels (as explained in Subsection 5.2). When these two error vectors overlap with each other, one will get a distortion of 3 pixels in a single block. Error vectors can modify two LSBs of pixels in each block, thus the maximum decimal difference between the original and the modified value of a pixel is equal to ±3. If maximum overlapping errors are present in every block, the values of the maximum MSE and minimum PSNR for a single block of 23 pixels are the same as for the entire 8-bit image, and are equal to:

$$\mathrm{MSE} = \frac{1}{23}\sum_{i=1}^{23}\left(p(1,i)-p'(1,i)\right)^2 = \frac{1}{23}\left(3^2+3^2+3^2\right) = \frac{27}{23} \tag{13}$$

$$\mathrm{PSNR} = 10\log\frac{255^2}{\mathrm{MSE}} \cong 47.434\mathrm{dB} \tag{14}$$

where $p(1, i)$ is an original value of the i-th pixel in a block, and $p'(1, i)$ is a modified value of the i-th pixel in a block.

The most undesired outcome (maximum MSE) for a 16-bit DICOM image occurs when the two 8-bit images (obtained from the original 16-bit DICOM image) contain maximum overlapping errors of 3 in each block of 23 pixels and additionally when errors from one 8-bit image overlap with errors from the second 8-bit image after reconstruction of the 16-bit DICOM image, as shown in Fig. 11.

In that case, error vectors can modify four LSBs of pixels in each block, thus the maximum decimal difference between the original and the modified value of a pixel is equal to ±15. Similarly as for the 8-bit image, the maximum MSE and minimum PSNR for the entire 16-bit DICOM image are equal to:

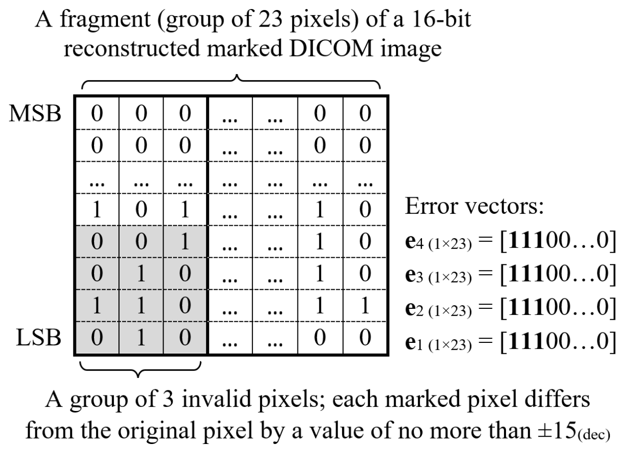

FIGURE 11. Example distribution of maximum overlapping errors inside a section (block of 23 pixels) of a 16-bit reconstructed marked DICOM image.

$$\mathrm{MSE} = \frac{1}{23}\sum_{i=1}^{23}\left(p(1,i)-p'(1,i)\right)^2 = \frac{1}{23}\left(15^2+15^2+15^2\right) = \frac{675}{23} \tag{15}$$

$$\mathrm{PSNR} = 10\log\frac{65535^2}{\mathrm{MSE}} \cong 81.654\mathrm{dB}. \tag{16}$$

It should be noted that reaching the minimum PSNR for a 16-bit DICOM image is extremely rare. Aside from the requirements mentioned in this subsection, four LSBs of all modified 16-bit pixels need to have either four 0s or four 1s to yield a maximum $\pm 15_{(\mathrm{dec})}$ change.

C. ANALYSIS OF MAXIMUM BPP FOR DICOM IMAGES

The bpp values were analysed for a decrypted 8-bit image with embedded data and a reconstructed 16-bit DICOM

TABLE 3. Embedding capacity for different RDHEI schemes, measured for a 16-bit DICOM image.

	Chen and Chang scheme [2]	[2] with the proposed binary decomposition	Proposed scheme with Golay (23, 12) code	Proposed scheme with Hamming (7, 4) code
bpp (8-bit)	-	0.42707	0.95651	0.85714
bpp (16-bit)	0.42707	0.85413	1.91302	1.71428

image with embedded data. Let us assume that the total number of pixels of a 16-bit DICOM image is a multiplication of 23, and let us assume that each block (group) of 23 pixels is used for data embedding. The maximum bpp for an 8-bit image is equal to 22/23 $\approx$ 0.95652 (as mentioned in Subsection 5.2), thus for the entire 16-bit DICOM image, the maximum bpp is equal to 44/23 $\approx$ 1.91304.

To compare our RDHEI method with the RDHEI method proposed by Chen and Chang [2], let us analyse the most optimistic scenario for [2], where every modifiable pixel can be reconstructed by neighboring pixels (there is no need for an error prediction map), thus all modifiable pixels are used for data embedding [2]. The maximum bpp is based on the image size. In general, the bigger the image, the more the bpp value approaches the value of 0.42857 (but will never reach it). The maximum value for the bpp derives from the use of Hamming (7, 4) code. The authors of [2] can embed up to 6 bits of additional data in every group of 7 modifiable pixels of the image. Based on the grid pattern of modifiable pixels (explained in [2]), and assuming (for simplicity) that the image is of size $N \times N$, the maximum bpp for [2] is equal to:

$$\text{bpp} = \frac{\left\lfloor \left\lceil \frac{1}{2}(N-2)^2 \right\rceil / 7 \right\rfloor \cdot 6}{N^2}. \tag{17}$$

If $N \to \infty$, then the upper limit for (17) is 6/7/2 $\approx$ 0.42857.

The Hamming (7, 4) code (which is a perfect code like Golay) can also be applied to our scheme. The main difference will be the length of the blocks. Instead of grouping 23 pixels we will be grouping 7 pixels for which we will be able to embed 6 bits of additional data (the syndromes of the codewords will be reduced from 11 bits to 3 bits). Such an approach will yield a worse maximum bpp, which is equal to 6/7 $\approx$ 0.85714 for an 8-bit image and 12/7 $\approx$ 1.71429 for the entire 16-bit DICOM image. However, it is easier to find smaller pixel blocks that are suitable for embedding when processing non-DICOM images.

An example comparison of bpp values for different RDHEI schemes is shown in Table 3. We used a random 16-bit DICOM image of size 1024 $\times$ 1280 px. The table presents bpp values for 8-bit images obtained from the 16-bit DICOM image and bpp values for the entire 16-bit DICOM image.

Please note that bpp results presented in Table 3 for the abovementioned image, as well as for any 16-bit DICOM image of monochrome photometric interpretation, will always be very close to the upper limit (maximum value, mentioned in this section) i.e. 1.91304 for a scheme with Golay (23, 12) code, 1.71429 for a scheme with Hamming (7, 4) code and 0.42857 for scheme [2] (provided we assume the most optimistic scenario for [2]). For the proposed scheme, a slight deviation of the presented values from the upper limit values derives from the last block, which is not used for data embedding if its size is not equal to 23 pixels for Golay (23, 12) code, or 7 pixels for Hamming (7, 4) code. Thus, the upper limit for bpp is only achievable if the image size is a multiple of 23 or 7.

D. ANALYSIS OF DATA EMBEDDING FOR NON-DICOM IMAGES

We analysed the embedding capabilities of the proposed scheme for $N = 20$ non-DICOM 8-bit images of different sizes. We implemented and compared the proposed method with Golay (23, 12) code and Hamming (7, 4) code. The results of the comparison are presented in Table 4.

We present the expected values with confidence intervals calculated from $N = 20$ simulations for each image size category. Each simulation was performed on a different image, randomly chosen from database [47]. Both schemes were analysed on the same set of images for each image size category. Due to the number of performed simulations, we used t-Student estimation [48]. The expected values presented in Table 4 are an estimation of the expected value μ, determined according to equation in [48]:

$$\mathrm{P}\left(\overline{X} - t_\alpha \frac{S}{\sqrt{N-1}} < \mu < \overline{X} + t_\alpha \frac{S}{\sqrt{N-1}}\right) = 1 - \alpha \tag{18}$$

where $\overline{X}$ is the expected value of the sample, S is the standard deviation of the sample, N is the size of the sample (20 simulations), t_α is a value obtained from the t-Student table for $N-1$ degrees of freedom, and α is the confidence coefficient equal to 5%.

Parameters "00..00/11..11 block count", presented in Table 4, show the number of applicable groups of 23 pixels (for Golay (23, 12) code) and 7 pixels (for Hamming (7, 4) code), used for data embedding (i.e. binary blocks of pixels where all MSBs were equal to 0 or 1).

Parameter $\text{bpp}_{\text{total}}$ indicates the total available embedding space, while parameter $\text{bpp}_{\text{usable}}$ refers to the available embedding space after embedding the permutation key (PK). It should be noted that the permutation key for the scheme with Golay (23, 12) code has a lesser impact (about a 7% decrease) on the embedding space than the permutation key for the scheme with Hamming (7, 4) code (about a 20% decrease). This property is derived from the larger pixel blocks which generate a shorter permutation key.

When embedding additional data in non-DICOM images, one can choose a scheme with either coding to achieve comparable embedding capacity. When applying Hamming

TABLE 4. Comparison of data embedding for non-DICOM images for the proposed scheme implemented with Golay (23, 12) code and Hamming (7, 4) code.

	256× 256 px		512× 512 px		1024× 1024 px	
Proposed scheme with Golay (23, 12) code	$\overline{X}$	$\frac{t_\alpha S}{\sqrt{N-1}}$	$\overline{X}$	$\frac{t_\alpha S}{\sqrt{N-1}}$	$\overline{X}$	$\frac{t_\alpha S}{\sqrt{N-1}}$
00..00 blocks count	8.156e+02	3.343e+02	3.909e+03	1.394e+03	1.763e+04	5.675e+03
11..11 blocks count	8.216e+02	3.083e+02	3.999e+03	1.360e+03	1.820e+04	5.735e+03
bpp_{total} (without PK)	0.54956	7.110e-02	0.66365	5.939e-02	0.75176	4.603e-02
bpp_{usable} (with embedded PK)	0.50609	7.110e-02	0.62018	5.939e-02	0.70828	4.603e-02
PSNR [dB]	54.333	6.212e-01	53.417	4.150e-01	52.832	2.783e-01
Proposed scheme with Hamming (7, 4) code	$\overline{X}$	$\frac{t_\alpha S}{\sqrt{N-1}}$	$\overline{X}$	$\frac{t_\alpha S}{\sqrt{N-1}}$	$\overline{X}$	$\frac{t_\alpha S}{\sqrt{N-1}}$
00..00 blocks count	3.690e+03	1.171e+03	1.589e+04	4.724e+03	6.674e+04	1.899e+04
11..11 blocks count	3.854e+03	1.182e+03	1.658e+04	4.785e+03	6.955e+04	1.917e+04
bpp_{total} (without PK)	0.69069	3.805e-02	0.74329	2.828e-02	0.77986	1.967e-02
bpp_{usable} (with embedded PK)	0.54784	3.805e-02	0.60043	2.828e-02	0.63701	1.967e-02
PSNR [dB]	50.560	2.529e-01	50.225	1.708e-01	50.009	1.113e-01

(7, 4) code for the proposed scheme, shorter blocks of 7 pixels (used for data embedding) are much easier to find than their longer counterparts. However, the situation changes the more unified the image hue is (which is often the case with large images with a blurred background). For such images, the proposed scheme with Golay (23, 12) code is often a better match. Additionally, application of Golay (23, 12) code introduces a shorter permutation key (which needs to be embedded), thus the bpp downgrade is always lesser than in the case of Hamming (7, 4) code.

VII. CONCLUSION

The proposed RDHEI scheme, intended for DICOM files, is a novel approach. The scheme is lossless, and based on the receiver's privileges, a recovery of marked data (with a high PSNR value), original data and embedded data is possible. When implemented inside a DICOM network, it allows for anonymized data storage outsourcing.

The results of the presented tests and analyses show that the proposed RDHEI method yields very high bpp and PSNR values for DICOM images, especially when compared with the method by Chen and Chang. The less complex structure, makes the proposed method well suited for real-time applications in the cloud. In addition, the proposed scheme can be successfully used for any non-DICOM files with an option to choose between Golay (23, 12) code or Hamming (7, 4) code to maximize the embedding capacity.

It should be noted that further research in the field of RDHEI can yield even more promising results. The proposed scheme can be further improved, e.g. by integrating RDHEI mechanisms with homomorphic encryption and applying it to the proposed modified DICOM network. In such a case, there will be no need for a fully homomorphic approach. An additive homomorphic encryption should be enough to perform anonymization and re-identification of encrypted DICOM files (without prior decryption) while maintaining unmodified tag information.

REFERENCES

[1] *Digital Imaging and Communications in Medicine (DICOM) Part 15: Security and System Management Profiles*, NEMA, Arlington, VA, USA, 2020.
[2] K. Chen and C.-C. Chang, "Real-time error-free reversible data hiding in encrypted images using (7, 4) Hamming code and most significant bit prediction," *Symmetry*, vol. 11, no. 1, pp. 1–17, 2019.
[3] Y.-Q. Shi, X. Li, X. Zhang, H.-T. Wu, and B. Ma, "Reversible data hiding: Advances in the past two decades," *IEEE Access*, vol. 4, pp. 3210–3237, 2016, doi: 10.1109/ACCESS.2016.2573308.
[4] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 3, pp. 354–362, Mar. 2006.
[5] X. Li, B. Li, B. Yang, and T. Zeng, "General framework to histogram-shifting-based reversible data hiding," *IEEE Trans. Image Process.*, vol. 22, no. 6, pp. 2181–2191, Jun. 2013.
[6] X. Li, W. Zhang, X. Gui, and B. Yang, "Efficient reversible data hiding based on multiple histograms modification," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 9, pp. 2016–2027, Sep. 2015.
[7] J. Tian, "Reversible watermarking by difference expansion," in *Proc. Workshop Multimedia Secur.*, 2002, pp. 19–22.
[8] A. M. Alattar, "Reversible watermark using the difference expansion of a generalized integer transform," *IEEE Trans. Image Process.*, vol. 13, no. 8, pp. 1147–1156, Aug. 2004.
[9] Y. Hu, H.-K. Lee, K. Chen, and J. Li, "Difference expansion based reversible data hiding using two embedding directions," *IEEE Trans. Multimedia*, vol. 10, no. 8, pp. 1500–1512, Dec. 2008.
[10] X. Li, J. Li, B. Li, and B. Yang, "High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion," *Signal Process.*, vol. 93, no. 1, pp. 198–205, Jan. 2013.
[11] X. Qu and H. J. Kim, "Pixel-based pixel value ordering predictor for high-fidelity reversible data hiding," *Signal Process.*, vol. 111, pp. 249–260, Jun. 2015.

[12] V. M. Manikandan and V. Masilamani, ''Reversible data hiding scheme during encryption using machine learning,'' *Procedia Comput. Sci.*, vol. 133, pp. 348–356, Jan. 2018.

[13] J. Mondal, D. Swain, and D. D. Panda, ''An improved RDH model for medical images with a novel EPR embedding technique,'' in *Advances in Computing and Data Sciences*. Singapore: Springer, 2018, pp. 421–430.

[14] K. Wong, K. Tanaka, K. Takagi, and Y. Nakajima, ''Complete video quality-preserving data hiding,'' *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 10, pp. 1499–1512, Oct. 2009.

[15] D. Xu, R. Wang, and Y.-Q. Shi, ''Reversible data hiding in encrypted H. 264/AVC video streams,'' *Lect. Notes Comput. Sci.*, vol. 23, no. 5, pp. 141–152, 2014.

[16] A. Nishimura, ''Reversible audio data hiding based on variable error-expansion of linear prediction for segmental audio and G.711 speech,'' *IEICE Trans. Inf. Syst.*, vol. E99.D, no. 1, pp. 83–91, 2016.

[17] F. Wang, Z. Xie, and Z. Chen, ''High capacity reversible watermarking for audio by histogram shifting and predicted error expansion,'' *Sci. World J.*, vol. 2014, May 2014, Art. no. 656251.

[18] K. Ma, W. Zhang, X. Zhao, N. Yu, and F. Li, ''Reversible data hiding in encrypted images by reserving room before encryption,'' *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 3, pp. 553–562, Mar. 2013.

[19] X. Zhang, ''Reversible data hiding in encrypted image,'' *IEEE Signal Process. Lett.*, vol. 18, no. 4, pp. 255–258, Apr. 2011.

[20] C. Qin and X. Zhang, ''Effective reversible data hiding in encrypted image with privacy protection for image content,'' *J. Vis. Commun. Image Represent.*, vol. 31, pp. 154–164, Aug. 2015.

[21] Z. Qian and X. Zhang, ''Reversible data hiding in encrypted images with distributed source encoding,'' *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 4, pp. 636–646, Apr. 2016.

[22] F. Huang, J. Huang, and Y.-Q. Shi, ''New framework for reversible data hiding in encrypted domain,'' *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 12, pp. 2777–2789, Dec. 2016.

[23] Y.-C. Chen, C.-W. Shiu, and G. Horng, ''Encrypted signal-based reversible data hiding with public key cryptosystem,'' *J. Vis. Commun. Image Represent.*, vol. 25, no. 5, pp. 1164–1170, Jul. 2014.

[24] D. Xu, K. Chen, R. Wang, and S. Su, ''Completely separable reversible data hiding in encrypted images,'' in *Proc. Int. Workshop Digit. Watermarking (IWDW)*, vol. 9569, Tokyo, Japan, 2016, pp. 365–377.

[25] X. Zhang, J. Long, Z. Wang, and H. Cheng, ''Lossless and reversible data hiding in encrypted images with public-key cryptography,'' *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 9, pp. 1622–1631, Sep. 2016.

[26] H.-T. Wu, Y.-M. Cheung, and J. Huang, ''Reversible data hiding in Paillier cryptosystem,'' *J. Vis. Commun. Image Represent.*, vol. 40, pp. 765–771, Oct. 2016.

[27] M. Li, D. Xiao, Y. Zhang, and H. Nan, ''Reversible data hiding in encrypted images using cross division and additive homomorphism,'' *Signal Process., Image Commun.*, vol. 39, pp. 234–248, Nov. 2015.

[28] D. Xu, K. Chen, R. Wang, and S. Su, ''Separable reversible data hiding in encrypted images based on two-dimensional histogram modification,'' *Secur. Commun. Netw.*, vol. 2018, Feb. 2018, Art. no. 1734961.

[29] X. Zhang, ''Separable reversible data hiding in encrypted image,'' *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 2, pp. 826–832, Apr. 2012.

[30] X. Wu and W. Sun, ''High-capacity reversible data hiding in encrypted images by prediction error,'' *Signal Process.*, vol. 104, pp. 387–400, Nov. 2014.

[31] Z. Qian, X. Zhang, and G. Feng, ''Reversible data hiding in encrypted images based on progressive recovery,'' *IEEE Signal Process. Lett.*, vol. 23, no. 11, pp. 1672–1676, Nov. 2016.

[32] P. Puteaux and W. Puech, ''An efficient MSB prediction-based method for high-capacity reversible data hiding in encrypted images,'' *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 7, pp. 1670–1681, Jul. 2018.

[33] F. Khelifi, ''On the security of a stream cipher in reversible data hiding schemes operating in the encrypted domain,'' *Signal Process.*, vol. 143, pp. 336–345, Feb. 2018.

[34] U.S. Department of Health & Human Services. *Summary of the HIPAA Security Rule*. Accessed: Apr. 19, 2021. [Online]. Available: https://www.hhs.gov/hipaa/for-professionals/security/laws-regulations/index.html

[35] Information Commissioner's Office. *Guide to the General Data Protection Regulation (GDPR)*. Accessed: Apr. 19, 2021. [Online]. Available: https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/

[36] OsiriX Development Team. *Technical Guides*. Accessed: Apr. 19, 2021. [Online]. Available: https://www.osirix-viewer.com/Documentation/Guides/index.html

[37] Medical Connections Ltd. *Products and Services*. Accessed: Apr. 19, 2021. [Online]. Available: https://www.medicalconnections.co.uk/

[38] D. Peck, ''Digital imaging and communications in medicine (DICOM): A practical introduction and survival guide,'' *J. Nucl. Med.*, vol. 50, no. 8, p. 1384, Aug. 2009.

[39] (Sep. 2012). *Philips CT Scanners and Workstations V2/V3 Revsion 7*. Accessed: Apr. 19, 2021. [Online]. Available: https://www.philips.pl/healthcare/resources/support-documentation/dicom-computed-tomography

[40] M. J. E. Golay, ''Notes on digital coding,'' *Proc. IRE*, vol. 37, p. 67, Jun. 1949.

[41] T. C. Lin, T. K. Truong, W. K. Su, P. Y. Shih, and G. Dubney, ''Decoding of the (24, 12, 8) extended Golay code up to four errors,'' *IET Commun.*, vol. 3, no. 2, pp. 232–238, 2009.

[42] C. Rorden. *Introduction to the DICOM Format*. Accessed: Apr. 19, 2021. [Online]. Available: https://web.archive.org/web/20150920230923/http:/www.mccauslandcenter.sc.edu/mricro/dicom/index.html

[43] A. Hewett, *CAR '97: Computer Assisted Radiology and Surgery, 11th International Symposium and Exhibition*. Berlin, Germany: Kuratorium OFFIS e.V., Jun. 1997.

[44] S. Barre. (2003). *Medical Image Samples*. Accessed: Apr. 19, 2021. [Online]. Available: https://web.archive.org/web/20160125103112/http:/stat.fsu.edu/pub/diehard/

[45] G. Marsaglia. (1995). *Diehard Battery of Tests of Randomness*. [Online]. Available: https://web.archive.org/web/20160125103112/, http://stat.fsu.edu/pub/diehard/

[46] CrypTool. (2020). *An Open-Source Windows Program for Cryptography and Cryptanalysis*. [Online]. Available: https://www.cryptool.org/en/

[47] Pexels. *Free Stock Photos*. [Online]. Available: https://www.pexels.com/

[48] P. Armitage, G. Berry, and J. N. S. Matthews, *Statistical Methods in Medical Research*. Chichester, U.K.: Wiley, 2008.

MARIUSZ DZWONKOWSKI received the M.Sc. and Ph.D. degrees from the Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, in 2011 and 2017, respectively. He is currently an Assistant Professor with the Medical University of Gdańsk, Poland, and the Gdańsk University of Technology, Poland. His research interests include cryptography, hyper-complex algebra, information coding, and reversible data hiding.

ROMAN RYKACZEWSKI received the M.Sc. and Ph.D. degrees from the Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, in 1968 and 1975, respectively. Since 1968, he has been an Academic Teacher with the Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology. His current research interests include cryptography, watermarking, steganography, and steganalysis.