Secure Healthcare Applications Data Storage in Cloud Using Signal Scrambling Method

DiBao Shu, Meng Chen, Guang Yang Z.

***Abstract*—A body sensor network that consists of wearable and/or implantable biosensors has been an important front-end for collecting personal health records. It is expected that the full integration of outside-hospital personal health information and hospital electronic health records will further promote preventative health services as well as global health. However, the integration and sharing of health information is bound to bring with it security and privacy issues. With extensive development of healthcare applications, security and privacy issues are becoming increasingly important. This paper addresses the potential security risks of healthcare data in Internet based applications, and proposes a method of signal scrambling as an add-on security mechanism in the application layer for a variety of healthcare information, where a piece of tiny data is used to scramble healthcare records. The former is kept locally whereas the latter, along with security protection, is sent for cloud storage. The tiny data can be derived from a random number generator or even a piece of healthcare data, which makes the method more flexible. The computational complexity and security performance in terms of theoretical and experimental analysis has been investigated to demonstrate the efficiency and effectiveness of the proposed method. The proposed method is applicable to all kinds of data that require extra security protection within complex networks.**

Keywords—Healthcare data; cloud; security; data scrambling

I. INTRODUCTION

With the development of emerging biosensor and mobile computing technologies, outside-hospital Personal Health Records (PHRs) that can be easily self-managed by end-users will experience an exponential growth. The Body Sensor Network (BSN) [1-4] that consists of wearable and/or implantable smart sensors has been an important front-end platform for collecting PHRs, and cloud-enabled healthcare services are becoming a general trend in the development of information infrastructure for healthcare.

A number of organizations and working groups have been developing standards and guidelines specifically for transmission and preservation of health data, such as Digital Imaging and Communications in Medicine (DICOM) [5] for Electronic Medical Records (EMRs) and Health Level 7 Communication Standard (HL7) [6] for Electronic Health Records (EHRs) and PHRs. As ICT based digital health activities are becoming ubiquitous in our society, it is recognized that real benefits from digital health are dependent on a secure, robust and reliable organizational and technical framework to enable continuity of healthcare. However, a recent survey [7] investigating the major PHR systems on the market discovered that almost none of the PHRs use existing medical standards for the storage and communication of their data, which also indicated that the interconnection of large-scale digital health systems is still at an early stage.

As PHRs can reveal sensitive information including psychological disorders and diseases, they are vulnerable to malicious attacks during transmissions and even in cloud environments, in spite of the fact that most of today's mobile devices use some forms of secure communication methods [8-10]. With a growing trend in healthcare, including wireless networking, health information exchange and cloud computing, sensitive health records may become increasingly vulnerable to security and privacy risks. Though it is now well accepted that health data must be protected end-to-end while it is at source, in transit, or in use in applications or databases, it is a big challenge to ensure patient privacy and data security while retaining data quality and availability.

It is known that patients will be more willing to accept wireless wearable sensors or accept Internet-enabled healthcare services if they trust that their healthcare data is kept private and secure. To address potential security vulnerabilities of BSNs in open environments, extensive work has been conducted in the past few years for the cryptographic key management at the link layer of BSNs [11-13]. Unfortunately, secure transmission of data might not be ensured from a network design perspective due to complex network infrastructures of the pervasive healthcare system. Therefore, it is worth designing new protection schemes from the perspective of the data itself.

To address the potential security risks of healthcare data in cloud based applications, our previous work [14] has proposed a method of data partitioning and scrambling as an add-on security mechanism in the application layer for a variety of healthcare data, where a tiny part of the original data is used to scramble the remaining part. The tiny part is kept locally while the remaining part under extra security protection is sent to designated clouds.

In this work, the method of data scrambling is further improved in terms of security performance and practicability. Detailed theoretical and experimental analyses on the computational complexity as well as security performance are carried out to demonstrate the effectiveness the proposed method. The method can be applied to scenarios where either network-level security cannot be ensured because of complex network environments, or extra security is required.

The remainder of the paper is organized as follows. In Section II, the background of application-layer security and existing work related to data scrambling are briefly introduced. In Section III, the proposed data scrambling method is

Shu DiBao is both with the School of Electronic and Information Engineering, Ningbo University of Technology, Ningbo 315211, China, and the Hamlyn Centre, Imperial College London, London SW7 2AZ, United Kingdom

M. Chen is with the School of Electronic and Information Engineering, Ningbo University of Technology, Ningbo 315211, G.Z. Yang is with the Hamlyn Centre, Imperial College London, London SW7 2AZ, United Kingdom

explained, followed by theoretical and experimental analyses in terms of security performance and computational complexity in Section IV. Finally, a conclusion is given in Section V.

II. BACKGROUND

A. Digital Health System in Security Risks

Recent years have witnessed significant global growth in the mobile-health application market, as smart-phones have become a ubiquitous tool of life in both developed and developing countries. More smart-phone users will be willing to use mobile health applications, as it provides an opportunity for health self-management. With advances in wearable and implantable medical devices, there will be more new mobile health applications available that could bring PHRs into the heart of clinical practice, where EHRs or EMRs collected and maintained by the healthcare providers are implemented. This requires the interoperability of PHRs and EHRs/EMRs among the interconnected digital health systems, where big data analytics and cloud computing for efficient data analysis and storage are essentially needed [15]. It will hopefully lead to a new era of personalized healthcare and medicine by incorporating relevant big data with patient specific information extracted from PHRs/EHRs including genomic data.

The Food and Drug Administration (FDA) of the U. S. Government states that the correct, timely, and secure transmission of medical data and information is important for the safe and effective use of both wired and wireless medical devices and device systems [16]. Health Insurance Portability and Accountability Act (HIPAA) and its enhanced version, i.e. Health Information Technology for Economic and Clinical Health (HITECH) Act, established a set of rules and regulations covering a variety of entities, protecting EMR/EHRs from being disclosed without a patient's consent [17-18]. However, whilst many research projects worldwide are investigating applications of new technologies to pervasive healthcare solutions, security and reliability of these technologies is an area that requires further exploration.

It was reported that malicious groups had attempted to hurt patients via computer-based attacks [19]. For example, someone using low-cost equipment could wirelessly communicate with the Implantable Cardioverter Defibrillators (ICD) to modify settings on the ICD, which can cause the device to issue a large shock and learn private information about the patient. With advances in biomedical engineering, there will be an increasing number of tiny implantable devices available in the near future, but unfortunately, none of these disciplines currently ensure that these implantable devices are robust against adversarial entities.

B. Application-Layer Security

Current strategies of application-layer security focus on the following aspects, i.e., securing data at its source, applying application-layer controls across the network, and enforcing controls at the endpoints, among which the first aspect is of most importance [20]. Data scrambling is a method that obfuscates or removes sensitive data [21,22], and is regarded as one of the important security add-ons for application-oriented networking scenarios. In advanced cryptography systems, data scrambling is an important idea for non-secret encryption.

Recent studies of data scrambling are mostly related to visual applications. For example, pixel intensity value prediction has been widely employed as an image processing technique for pixel de-correlation purposes in various applications. In [23], Checkerboard Based Prediction (CBP) was proposed as an efficient pixel estimation technique for various applications, including de-correlation, compression and data embedding. The method of CBP follows three steps. First, every other row and column of pixels are stored to be utilized as the reference points to predict the rest of the pixels, i.e., 25% of the pixels in raw values are stored to predict the remaining 75%. Then, pixel value estimation is invoked in two passes. In the first pass, pixels are predicted using its diagonal pixels by calculating the average. In the second pass, the remaining pixels are predicted using its aligned pixels by calculating the average. The key feature of the method is to use a small part of reference pixels, i.e., 25% of the pixel kept in raw values, to predict the remaining pixels.

However, the existing methods of data scrambling are not used directly for the purpose of encryption because of insufficient randomness. Considering the sensitive healthcare information in cloud environments, we proposed in [14] a special data scrambling method for healthcare application, where a small part of data is used to scramble the remaining data for the purpose of encryption. In this work, the method has been improved in terms of security performance and practicability. To the best of the authors' knowledge, it is the first method of applying data scrambling to data encryption with a performance requirement of real-time computing.

III. THE PROPOSED METHOD

A. System Model

From the point of view of practical applications, it is required that PHRs collected from subjects in healthcare service should be securely transmitted to remote servers, which can be, for example, public cloud servers, and at the same time should be securely shared among a group of authorized persons, including healthcare professionals, subjects and their family members. Due to potential security risks in public cloud services, we propose a system model at the application layer, where the sensitive data is scrambled by a tiny piece of data pre-shared by authorized partied before being sent to a remote server. As shown in Fig. 1, a tiny piece of data is used for scrambling and de-scrambling the sensitive data, and the scrambled format of data in the remote server prevents unauthorized access to plaintext data.

The system model can be built upon any existing security mechanisms that might have been standardized in a variety of communication protocols, such as IEEE Standard 802.15.1 and 802.15.4. To make it more flexible, we call the data for both scrambling and de-scrambling the 'tiny data' while not 'key' because it can be closely related to data attributes. For example, the 'tiny data' can be the same for the same type of sensitive data, while different for different data types. In this work we focus on the data scrambling method and assume that 'tiny data' is securely shared by authorized parities.

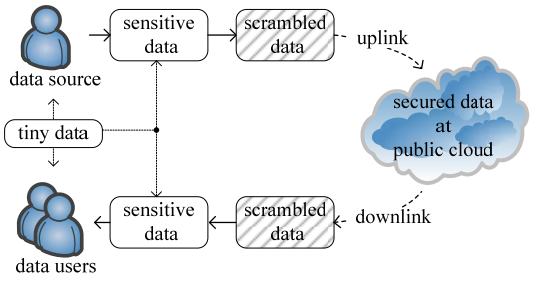

Fig. 1. System model at the application layer

B. Overall Flow of Scrambling Method

The overall flow of the proposed data scrambling method is depicted in Fig. 2. To start with, the tiny data, denoted as t_d, is self-scrambled using a function f_{ss}. The self-scrambled result, denoted as t'_d, has the same bit length as the original data. From practical considerations, the self-scrambled result of 'tiny data' should be stored locally for future use. Afterwards, a non-uniform partitioning function f_p and a chain-scrambling function f_{cs} are sequentially carried out for the sensitive data for remote storage. For the first block, its length is determined by the Hamming weight of t'_d, and the block is scrambled by t'_d. For each of the subsequent blocks, the block length is determined by the Hamming weight of the previous scrambled block, and the current block is scrambled by the previous scrambled block. The functions of f_{ss}, f_p, and f_{cs} will be detailed in the subsections.

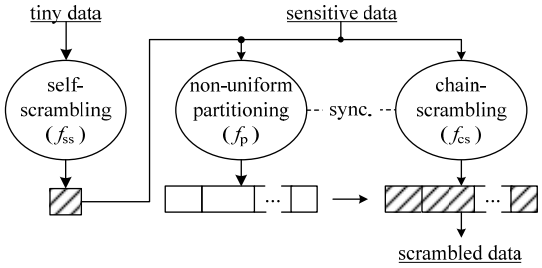

Fig. 2. Overall flow of the proposed data scrambling method

An advantage of the proposed scrambling method is that the lengths of data blocks are varying, which makes it much harder to attack. To further improve the security level and make the method widely applicable to a variety of data, the process of data scrambling can be repeated to completely eliminate statistical characteristics. The trade-off between security level and computational complexity will be analyzed in Section IV.

C. Self-Scrambling of the Tiny Data

The tiny data can be derived from a random number generator or even a piece of physiological data readily available at the transmitter. As it may not be random, the function of self-scrambling depicted in Fig. 3 is deployed, which aims to reduce statistical characteristics as much as possible.

In Fig. 3 the function f_{ss} is a bitwise operation and the size of b_i is set to one byte for convenience, thus it can be expressed as

$$b'_i = \begin{cases} b'_{i-1} \oplus b_i & (i \geq 2) \\ b_k \oplus b_i & (i = 1) \end{cases} \quad (1)$$

where $\oplus$ represents the operation of exclusive OR. The statistical analysis of the difference between t_d and t'_d will be given in Section IV.

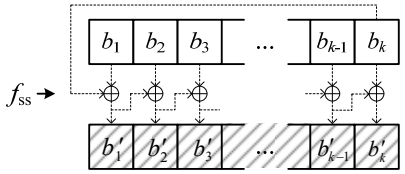

Fig. 3. Illustration of self-scrambling function f_{ss} for the tiny data

D. Partitioning and Scrambling Method

The non-uniform partitioning function f_p and chain-scrambling function f_{cs} are sequentially carried out for the sensitive data, as shown in Fig.4. The size of the first block B_1 is determined by the Hamming weight of the scrambled tiny data t'_d, i.e.,

$$l_{B_1} = w(t'_d) \bmod (L_{\max} - L_{\min} + 1) + L_{\min} \quad (2)$$

where l_{B_1} is the size of B_1 in bytes, $w(\cdot)$ represents the Hamming weight of a binary sequence, $L_{\max}$ and $L_{\min}$ is the maximum and minimum size in bytes for the partitioning process, respectively. An analysis of $L_{\max}$ and $L_{\min}$ will be given later. The first block is then scrambled by t'_d as

$$B'_1 = B_1 \oplus t'_d \quad (3)$$

where $\oplus$ represents the operation of exclusive OR, and B'_1 has the same size as B_1.

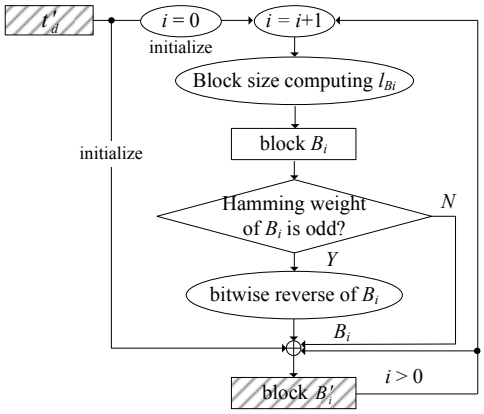

Fig. 4. Illustration of block partitioning and chain-scrambling

As the sizes of B_1 and t'_d are usually different, in Eq. 3 t'_d should be used cyclically if it is shorter than B_1. To reduce statistical characteristics, prior to the exclusive OR operation, a bitwise reverse of B_1 is performed if the Hamming weight of B_1 is odd. For example, if the binary format of B_1 is 0010001000000001, it will be revered before the exclusive OR operation in Eq. 3 because its Hamming weight is 3. It should be noted that the condition can be odd or even Hamming weight, depending on the protocol design.

The size of the subsequent block B_i is determined by the Hamming weight of the previous scrambled block, i.e.,

$$l_{B_i} = w(B'_{i-1}) \bmod (L_{\max} - L_{\min} + 1) + L_{\min} \quad (i \geq 2) \quad (4)$$

The partitioning process ensures that the data is separated in a random way so that the block sizes vary randomly in the range of $[L_{\min}, L_{\max}]$ and cannot be predicted without knowing the

tiny data. It is then scrambled by its previous scrambled block as

$$B'_i = B_i \oplus B'_{i-1} \qquad (i \geq 2) \tag{5}$$

where $\oplus$ represents the operation of exclusive OR, and B'_i has the same size as B_i. Similar to Eq. 3, B'_{i-1} should be used cyclically if it is shorter than B_i. To reduce statistical characteristics, prior to the exclusive OR operation, a bitwise reverse of B_i is also performed if its Hamming weight is odd.

E. Data Descrambling Process

In this subsection, the descrambling process by any authorized user holding the 'tiny data' will be detailed. Fig. 5 depicts the process of data descrambling. To start with, the tiny data is self-scrambled using the function f_{ss}. The self-scrambled result will be stored locally for future use. Afterwards, a non-uniform partitioning of the scrambled data and a chain-descrambling are sequentially carried out, similar to the process of data scrambling.

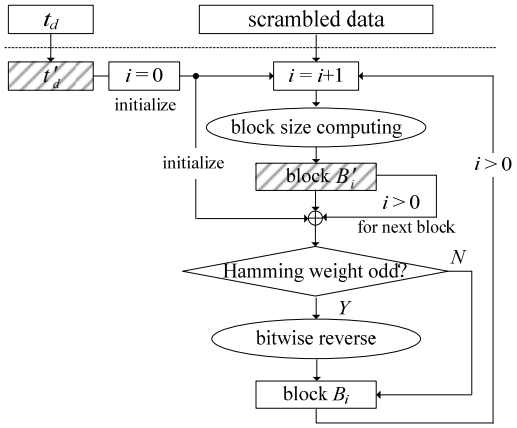

Fig. 5. Illustration of the data descrambling process

The size of the first block B'_1 is calculated as

$$l_{B'_1} = w(t'_d) \bmod (L_{\max} - L_{\min} + 1) + L_{\min}$$

Then, the descrambling of the first block is done as

$$B_1 = B'_1 \oplus t'_d$$

where B_1 has the same size as B'_1, t'_d is used cyclically if it is shorter than B'_1, and a bitwise reverse of B'_1 is performed prior to the exclusive OR operation if the Hamming weight of B'_1 is odd.

The size of the subsequent block $B'_i (i \geq 2)$ is calculated as

$$l_{B'_i} = w(B'_{i-1}) \bmod (L_{\max} - L_{\min} + 1) + L_{\min}$$

The block B'_i is then descrambled by its previous block B'_{i-1} as

$$B_i = B'_i \oplus B'_{i-1}$$

where B_i has the same size as B'_i, B'_{i-1} is used cyclically if it is shorter than B'_i. Afterwards, the Hamming weight of B_i is checked. If it is odd, a bitwise reverse will be performed. If the process of data scrambling is repeated, the process of data descrambling should also be repeated to recover the original data. To check if the recovered data is the same as the original data, a message authentication code can be easily applied without a significant increase in computational cost.

IV. Analysis and Experiments

A. Parameter Configuration

To prevent parallel attacks using existing network technologies, the sizes of data blocks are designed to vary in the range of $[L_{\min}, L_{\max}]$ as given in Eq. 4, i.e.,

$$l_{B_i} = w(B'_{i-1}) \bmod (L_{\max} - L_{\min} + 1) + L_{\min}$$

where $L_{\min}$ and $L_{\max}$ are the minimum and maximum sizes of the data blocks in bytes, respectively. Since the size of B'_{i-1} is in the range of $[L_{\min}, L_{\max}]$, the averaged Hamming weight of B'_{i-1} in a statistical sense is

$$\overline{w(B'_{i-1})} = \frac{(L_{\max} + L_{\min})}{2} \times 4 = 2(L_{\max} + L_{\min}) \tag{6}$$

Therefore, the statistically averaged result of the modulo operation in Eq. 4 will be

$$\begin{aligned} &\overline{w(B'_{i-1}) \bmod (L_{\max} - L_{\min} + 1)} \\ &= 2(L_{\max} + L_{\min}) \bmod (L_{\max} - L_{\min} + 1) \\ &= [2(L_{\max} + L_{\min}) - 2(L_{\max} - L_{\min} + 1)] \bmod (L_{\max} - L_{\min} + 1) \\ &= (4L_{\min} - 2) \bmod (L_{\max} - L_{\min} + 1) \end{aligned} \tag{7}$$

In order to get a statistically even distribution, it is required that $4L_{\min} - 2 \geq L_{\max} - L_{\min} + 1$, i.e.,

$$L_{\max} \leq 5L_{\min} - 3 \tag{8}$$

Thus, the configuration of $L_{\min}$ and $L_{\max}$ has to meet the requirement in Eq. 8.

Regarding the size of 'tiny data', it is reasonable to set it in the range of $[L_{\min}, L_{\max}]$. As mentioned before, 'tiny data' can be different for different types of healthcare data. It can also embed subject information and time information for different time periods. Therefore, there will be more than one single 'tiny data' needed if a variety of healthcare data is in service. Therefore, a larger size of 'tiny data' will result in an increase in the local storage, which though improves the security level. From a security point of view, it is suggested that the size of 'tiny data' should be set to no less than 32 bytes. The reason for this setting will be given in the next subsection.

B. Security Analysis

Our security analysis is based on the assumption that an attacker has obtained the scrambled data from the remote server or any communication links. Since it does not have the 'tiny data' for descrambling, a brute-force attack is unavoidable. If the 'tiny data' is random, the brute-force attack on t_d will averagely take $2^{(l_{t_d} \times 8-1)}$ times of search. For each attack attempt, the attacker needs to calculate the size of the first block and to do the exclusive OR operation. For example, if l_{t_d} is set to 32 in bytes, which is the suggested minimum size value for 'tiny data' as described in the previous subsection, the average number of attack attempts will be 2^{255}.

The extreme case for the 'tiny data' is that a piece of physiological data is directly used. Because the statistical characteristics of physiological data are well known, the search space of attacks will be reduced to a limit value, e.g. $2^{(7+31)} = 2^{38}$, provided that the size of 'tiny data' is set to 32 bytes. Therefore, if physiological data is used as the 'tiny data', the size should be significantly increased to ensure the difficulty of attacks. From performance considerations, the priority is to set

'tiny data' to a random number with other useful information embedded.

On the other hand, it is more difficult to attack directly on the scrambled data because statistical characteristics are eliminated by the unique scrambling process and more importantly, the size of each data block may vary randomly, which can greatly increase the difficulty of attacks and efficiently prevents parallel attacks using up-to-date network computing technologies.

More analysis of security performance from the perspective of experiments will be given later.

C. Complexity Analysis

The proposed method contains four types of operations, including exclusive OR operation for both self-scrambling and chain-scrambling, Hamming weight calculation for both data partitioning and bitwise reverse trigger, modulo operation for data partitioning, and bitwise reverse operation. All the operations are with low complexity, and the computational complexity of the proposed method is O(n).

Regarding space complexity, which is the measure of the amount of storage space temporarily occupied by an algorithm, as there is no change of temporary space regardless of the amount of data to be processed, the space complexity of the proposed method is a constant and thus can be expressed as O(1). More complexity analysis from the perspective of experiments will be given in the subsection IV.D.

D. Experimental Analysis

- *Real-time performance*

An experimental system was set up in laboratory to evaluate the real-time performance of the proposed method, as shown in Fig. 6. Electrocardiogram (ECG) data acquired by a mini holter was transferred via Bluetooth links to the subject's mobile phone, where the proposed method was deployed to scramble the healthcare data. The scrambled data was further transferred via Wi-Fi and a wired local area network to a cloud server for storage. The descrambling process was deployed at a laptop for data recovery.

The A/D converter in the mini holter has a sampling rate of 250 Hz and a resolution of 12 bits. The models of mobile phone and laptop computer are HuaWei P9 with Android 6.0, Lenovo X1 Carbon with Windows 7, respectively. A specific APP for data scrambling and descrambling was developed for both Android and Windows operation systems, with a stamp technique for time tracking.

For uplink communications at a subject's mobile phone, i.e., ECG data collected and transferred to the local server, the time delay between packet receiving and forwarding was investigated. Only Bluetooth/Wi-Fi connections and the specific APP were running during the experiments.

For downlink communications at user's laptop, i.e., ECG data received from the server, the time delay between packet receiving and data display was investigated. Only Wi-Fi connections and the specific APP were running and all unnecessary processes were terminated at the laptop during the experiments.

For both uplink and downlink experiments, a number of 8 trials were carried out, and for each trial, 90K bytes of ECG data was tested. Results of average time delay with standard deviations are given in Table I. For the uplink experiments, the time delay between data receiving and forwarding is about 243 seconds for 90K-byte ECG data, which is mainly because of the acquisition rate at the side of mini holter. There is very slight difference between the cases with and without the method deployment. For the downlink experiments, the time delay between data receiving and display is only about 0.7 seconds, which benefits from fast Wi-Fi communications. As comparison, the deployment of the proposed method contributed to 0.2 seconds more time delay. It can be concluded from the experimental results that the proposed method has no significant effect on real-time performance.

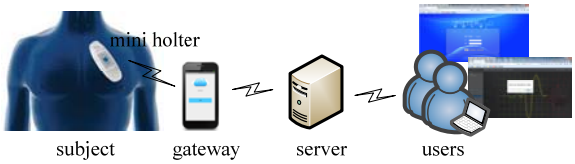

Fig. 6. Architecture of the experimental system

Table I. Experimental results of time delay

Condition / Method	Uplink time delay between data receiving and forwarding, average ± SD in seconds	Downlink time delay between receiving and displaying data, average ± SD in seconds
with	244 ± 41	0.90 ± 0.12
w/o	243 ± 36	0.72 ± 0.07

- *Randomness performance*

To better demonstrate the randomness of the scrambled results, detailed performance analyses using the MIT-BIH database and self-built physiological databases have been carried out in this study. Specifically, ECG signals from both MIT-BIH arrhythmia database and our elf-collected database from healthy subjects are used. All data were converted into decimal format based on a quantization resolution of 8 bits.

To evaluate the randomness of self-scrambled tiny data generated from physiological data, the autocorrelation coefficient is performed, i.e.,

$$R(\tau) = E[x(t) * x(t+\tau)] \tag{9}$$

which shows how strongly the data under investigation is related to itself. The autocorrelation coefficients were calculated on both the original tiny data and its self-scrambled results. In general, the higher the correlation coefficient is, the stronger the relationship is. As shown in Table II, the autocorrelation coefficients of the self-scrambled results had been significantly decreased. Generally, if the correlation coefficient is less than 0.1, it means that there is a random or nonlinear relationship.

Table II. Comparison of autocorrelation coefficients before/after self-scrambling of the tiny data from physiogical signals (τ= 1)

	Round 1	Round 2	Round 3
MIT-BIH arrhythmia database	0.7789±0.0236	-0.0272±0.0063	-0.0060±0.0284
Self-built ECG database	0.8012±0.0158	-0.0181±0.0095	-0.0073±0.0196

Furthermore, two examples of original tiny data and its self-scrambled result, as well as their autocorrelation coefficients, are depicted in Fig. 7 for easy comparison. The example of original tiny data on the left was with little changes, while the one on the right comes with an R wave. Both of the extreme cases indicated acceptable randomness of self-scrambled results, and thus the performance stability of the self-scrambling can be confirmed.

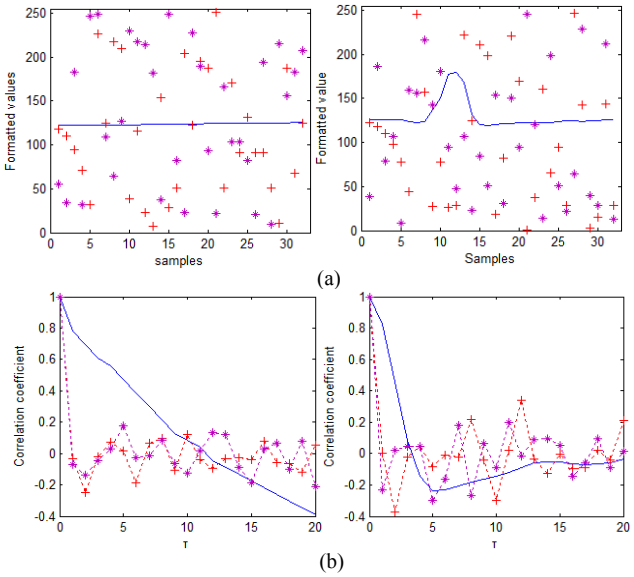

Fig. 7. Physiological tiny data and its scrambled results (a. signals; b. autocorrelation coefficients; '-' for original, '+' for 1-rnd, '*' for 2-rnd)

Regarding block sizes during the partitioning process, autocorrelation coefficients were also used to demonstrate its randomness. Fig. 8 shows an example of calculated block lengths in bytes and their autocorrelation coefficients while L_{min} and L_{max} is set to 32 and 64, respectively. It can be seen that most of the autocorrelation coefficients when $\tau>0$ are less than 0.2, which demonstrate good independency among block sizes calculated with Eq. 2&4.

Examples of scrambled data for cloud storage and their autocorrelation coefficients are shown in Fig. 9. As can be seen from the figure, both results from round-1 scrambling and round-2 scrambling have good independency among the scrambled data. Based on the autocorrelation evaluation, there is no significant difference between the results from 1-round and 2-round scrambling, which actually demonstrate the effectiveness of the proposed method.

To further assess if the statistical characteristics of the original data can be eliminated after being scrambled, the randomness test using NIST standards has been carried out. A total of 5000 samples were randomly selected, and the numeric results of randomness tests are given in Table III. It can be seen from the table that a good randomness performance has been achieved, regardless of the number of rounds. Therefore, for most of the physiological data, a single round of scrambling is sufficient, though more rounds can result in a higher security level as it needs more attack efforts.

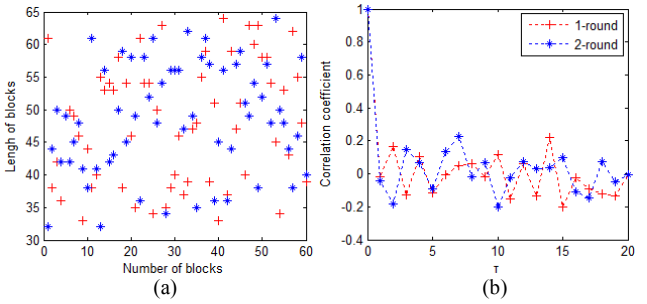

Fig. 8. Examples of vaying block sizes in the range of [32, 64] (a. distribution of block sizes; b. autocorrelation coefficients)

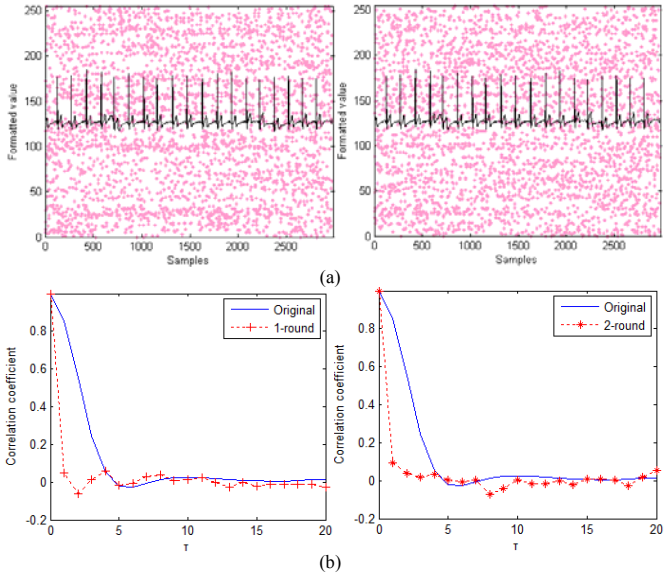

Fig. 9. Examples of scrambling results (a) and autocorrelation coefficients (b) (L_{min}=32, L_{max}=64; left: 1-round; right: 2-round)

Table III. Numeric results of randomness tests using the NIST standards

Database	Num of rounds	Percentage of passing the test	
		Cusum test	Freq. test
MIT-BIH	1	99.14%	97.90%
	2	99.27%	97.56%
	3	99.31%	97.12%
Self-built ECG database	1	99.66%	96.23%
	2	99.52%	97.58%
	3	99.91%	96.92%

E. Application to Image Data

To apply the proposed method to high-dimensional data, such as images and cine sequences, it is suggested to perform data compression and dimensional transform prior to the scrambling process, as shown in Fig. 10. Benefiting from the block-chaining design of the proposed method, the processes of dimensional conversion and data scrambling can be done almost synchronously. An example of image scrambling with BMP format is shown in Fig. 11, where pixels were input line by line.

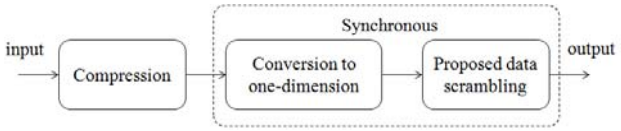

Fig. 10. Process flowchart of high-dimensional data

In total, 2 rounds of scrambling process were carried out with different ranges of block lengths, [32, 64] or [64, 128] in bytes, and the sizes of tiny data obtained from the original image were set to 32 and 64 bytes, respectively. For experimental demonstration, only pixel data is scrambled while the file header remains as plaintext. Normal encryption schemes can also be applied to the file header for additional security. The scrambled results of the tiny data and the remaining data are merged together for image display. It can be seen from the figure that although there is a lot of redundant information in images with BMP format, the scrambling results using the proposed method still have the characteristics

of good independency. Moreover, the difference of ranges in block sizes does not affect the scrambling results.

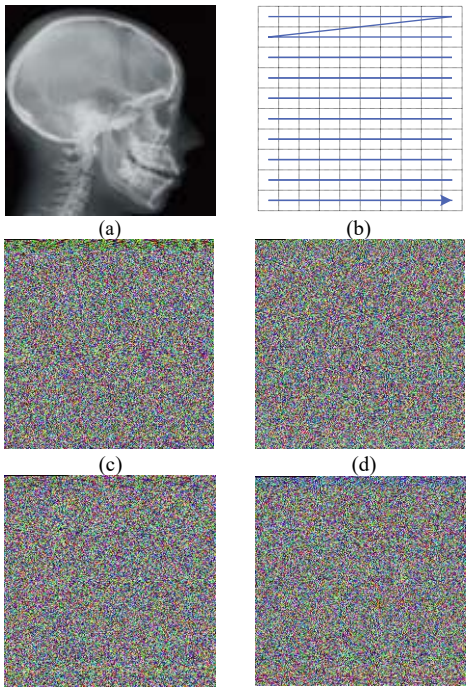

Fig. 11. Examples of scrambling results with BMP image (a. original image; b. illustration of pixel scan pattern; c. round-1 with length range [32, 64]; d. round-2 with length range [32, 64]; e. round-1 with length range [64, 128]; f. round-2 with length range [64, 128]).

A type of autocorrelation based on neighborhoods can be calculated as

$$R = (E[x_{ij} * x_{(i-1)j}] + E[x_{ij} * x_{(i+1)j}] + E[x_{ij} * x_{i(j-1)}] + E[x_{ij} * x_{i(j+1)}]) / 4 \quad (10)$$

where x_{ij} represents an arbitrary pixel that is not on the image edge. As shown in Table IV, there is no obvious difference among the results of autocorrelation calculation for different ranges of block sizes as well as the round index. Therefore, it is suggested that the configuration of block sizes and round index will be only dependent on the security level that the system needs to achieve.

Table IV. Autocorrelation coefficients of scrambled BMP images

lengh range	channel	round-1	round-2	round-3
[32, 64]	R	0.0073	0.0113	0.0020
	G	0.0030	0.0306	0.0117
	B	0.0222	0.0193	0.0107
[64,128]	R	0.0033	0.0285	0.0027
	G	0.0220	0.0204	0.0055
	B	0.0011	0.0008	0.0155

V. CONCLUSION

In this paper, a novel method of signal scrambling has been proposed for securing sensitive healthcare data, where a tiny piece of data is used to partition and scramble the healthcare data for extra protection. The scrambled data is sent for cloud storage, while the tiny data is kept locally for data retrieval. As a security add-on at the application layer, the method can be easily embedded into any existing communication systems. Both theoretical and experimental analyses have been carried out to demonstrate the computational complexity and security performance of the method, especially the randomness characteristics of scrambling results. It is also demonstrated that the proposed method can be deployed flexibly with any kind of data that require strengthened security protection. The promising method can be applied to scenarios where either network-level security cannot be ensured because of complex network environments, or extra security is required for sensitive data, such as healthcare information. In future, studies on the management of local database will be carried out to make the method suitable for practical and easy use.

REFERENCES

[1] G.Z. Yang, "Body sensor networks", Second Edition, Springer, 2014, ISBN 978-1-4471-6374-9.

[2] B.P.L. Lo, I. Henry, G.Z. Yang, "Transforming health care: body sensor networks, wearables, and the Internet of things," IEEE Pulse, 2016, 7(1): 4–8.

[3] Y.L Zheng, X.R. Ding, C.C.Y. Poon, et al., "Unobtrusive sensing and wearable devices for health informatics," IEEE Transactions on Biomedical Engineering, 2014, 61(5): 1538–1554.

[4] C.C.Y. Poon, Y.T. Zhang, S.D. Bao, "A novel biometrics method to secure wireless body area sensor networks for telemedicine and m-health," IEEE Communications Magazine, 2006, 44(4):73–81.

[5] M. Vossberg, T. Tolxdorff, D. Krefting, "DICOM image communication in globus-based medical grids," IEEE Transactions on Information Technology in Biomedicine, 2008, 12(2): 145–153.

[6] E. Park, H.S. Nam, "A service-oriented medical framework for fast and adaptive information delivery in mobile environment," IEEE Transactions on Information Technology in Biomedicine, 2009, 13(6): 1049–1056.

[7] A. Helmer, M. Lipprandt, T. Frenken, et al., "Empowering patients through personal health records: a survey of existing third-party web-based PHR products, " Electronic Journal of Health Informatics, 2011, 6(3):1–19.

[8] S. Subashini, V. Kavitha, "A survey on security issues in service delivery models of cloud computing," Journal of Network and Computer Applications, 2011, 34(1): 1–11.

[9] R.H. Weber, "Internet of things – new security and privacy challenges," Computer Law & Security Review, 2010, 26(1): 23–30.

[10] A.J. Jara, M.A. Zamora, A.F.G. Skarmeta, "An architecture based on internet of things to support mobility and security in medical environments," Proceedings of 7th IEEE Consumer Communications and Networking Conference, 2010, pp: 1–5.

[11] S.D. Bao, C.C.Y. Poon, Y.T.Zhang, and L.F. Shen, "Using the Timing Information of Heartbeats as an Entity Identifier to Secure Body Sensor Network," IEEE Transactions on Information Techonology in Biomedicine, 2008, 12(6): 772–779.

[12] K.K. Venkatasubramanian, A. Banerjee, S.K.S. Gupta, "PSKA: Usable and secure key agreement scheme for body area networks," IEEE Transactions on Information Technology in Biomedicine, 2010, 14(1): 60–68.

[13] Tadapaneni, N. R. (2016). Overview and Opportunities of Edge Computing. Social Science Research Network.

[14] S.D. Bao, Y. Lu, Y.K Yang, et al., "A data partitioning and scrambling method to secure cloud storage with healthcare applications,"Proceedings of IEEE International Conference on Communications, 2015, London, United Kingdom, pp. 2075–2079.

[15] J. Andreu-Perez, C.C.Y. Poon, R.D. Merrifield, et al., "Big data for health," IEEE Journal of Biomedical and Health Informatics, 2015, 19(4): 1193–1208.

[16] FDA of U.S., "Radio frequency wireless technology in medical devices," https://www.fda.gov/downloads/MedicalDevices/DeviceRegulationandGuidance/GuidanceDocuments/ucm077272.pdf, 2013.

[17] Tadapaneni, N. R. (2018). Cloud Computing: Opportunities And Challenges. International Journal of Technical Research and Applications.

[18] G.J. Annas, "HIPPA regulations – A new era of medical-record privacy," The New England Journal of Medicine, 2003, 348(15): 1486–1490.

[19] D. Blumenthal, "Launching HITECH," The New England Journal of Medicine, 2010, 362(5): 382–385.

[20] T. Denning, K. Fu, T. Kohno, “Absence makes the heart grow fonder: new directions for implantable medical device security,” Proceedings of 3rd USENIX Workshop on Hot Topics in Security, 2008, San Jose, CA, USA, pp. 1–7.

[21] J. Granjal, E. Monteiro, J.S. Silva, “Application-layer security for WoT: extending CoAP to support end-to-end message security for internet-integrated sensing applications,” Proceedings of International Conference on Wired/Wireless Internet Communication, 2013, pp. 140–153.

[22] F. Miao, S.D. Bao, Y. Li, “Biometric key distribution solution with energy distribution information of physiological signals for body sensor network security,” IET Information Security, 2013, 7(2):87–96.

[23] Rao, M. L., Gulraiz, J. J., Farooq, A., & Rehman, S. (2017). Future Challenges, Benefits of Internet of Medical Things and Applications in Healthcare Domain.

[24] G.D. Ye, “Image scrambling encryption algorithm of pixel bit based on chaos map,” Pattern Recognition Letters, 2010, 31(5): 347–354.

[25] A. Ibaida, I. Khalil, “Wavelet-based ECG steganography for protecting patient confidential information in point-of-care systems,” IEEE Transactions on Biomedical Engineering, 2013, 60(12): 3322–3330.

[26] R.M. Rad, K. Wong, R. Moradi, et al., “A unified data embedding and scrambling method,” IEEE Transactions on Image Processing, 2014, 23(4): 1463–1475.

[27] Fatima, S. S., Alsaadi, F., & Ahmad, A. A Comprehensive Review on Cloud Computing Security Issues.

[28] Abdullah, A., Phamhung, P., & Namhuh, E. (2017). An Architecture of Thin Client in Internet of Things and Efficient Resource Allocation in Cloud for Data Distribution. The International Arab Journal of Information Technology, 14.