Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/MBE

MBE, 16(5): 3367–3381
DOI: 10.3934/mbe.2019168
Received: 22 January 2019
Accepted: 04 March 2019
Published: 19 April 2019

Research article

Algebraic secret sharing using privacy homomorphisms for IoT-based healthcare systems

Ching-Chun Chang[1,*] **and Chang-Tsun Li**[2]

[1] Department of Computer Science, University of Warwick, Coventry, United Kingdom
[2] School of Information Technology, Deakin University, Geelong, Australia

* **Correspondence:** Email: ching-chun.chang@warwick.ac.uk.

Abstract: Healthcare industry is one of the promising fields adopting the Internet of Things (IoT) solutions. In this paper, we study secret sharing mechanisms towards resolving privacy and security issues in IoT-based healthcare applications. In particular, we show how multiple sources are possible to share their data amongst a group of participants without revealing their own data to one another as well as the dealer. Only an authorised subset of participants is able to reconstruct the data. A collusion of fewer participants has no better chance of guessing the private data than a non-participant who has no shares at all. To realise this system, we introduce a novel research upon secret sharing in the encrypted domain. In modern healthcare industry, a patient's health record often contains data acquired from various sensor nodes. In order to protect information privacy, the data from sensor nodes is encrypted at once and shared among a number of cloud servers of medical institutions via a gateway device. The complete health record will be retrieved for diagnosis only if the number of presented shares meets the access policy. The retrieval procedure does not involve decryption and therefore the scheme is favourable in some time-sensitive circumstances such as a surgical emergency. We analyse the pros and cons of several possible solutions and develop practical secret sharing schemes for IoT-based healthcare systems.

Keywords: healthcare systems; homomorphic encryption; Internet of things; secret sharing

1. Introduction

Internet of Things (IoT) is an emerging technology that utilises cloud connected devices to collect data for analysis. Healthcare industry is one of the most promising fields that have adopted IoT solutions since its early stage. The development of wearable technology, wireless body area network and cloud computing has established a new way for medical practitioners to acquire health data from patients. It greatly benefits health monitoring, epidemiological studies, and pharmaceutical re-

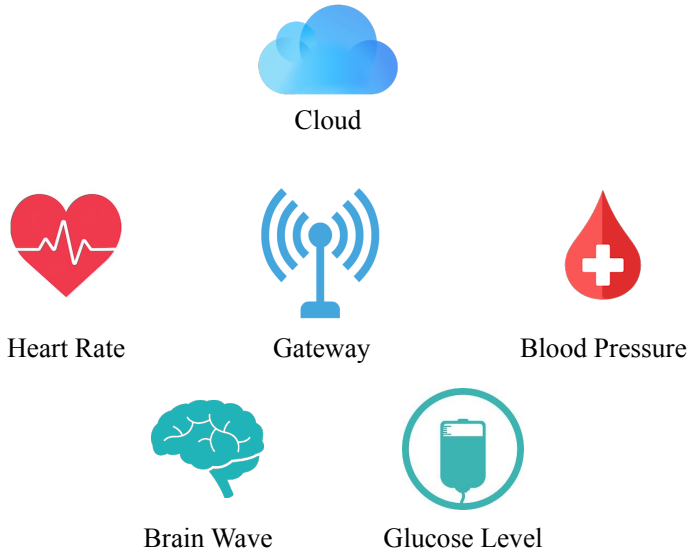

Figure 1. An IoT-based healthcare architecture. The health data acquired from the sensor nodes (*e.g.* heart rate, blood pressure, brain wave, and glucose level) is aggregated at the gateway and store in the cloud.

search [1–5]. A common IoT-based architecture for healthcare applications is illustrated in Figure 1, which consists of a gateway device, a cloud server and several sensor nodes. Each sensor node can be viewed as a wearable equipment used for monitoring the health status of an individual, such as heart rate, blood pressure, brain wave, glucose level, *etc*. Under the given framework, the sensor nodes send the medical data to a local gateway device via wireless communication such as Wi-Fi or Bluetooth, whereas the gateway device aggregates the data and store it in the cloud server for further analysis. However, there are risks of information leakage during data transmission and storage. For example, an adversary may attempt to eavesdrop the wireless communication, attack the gateway device or even access to the cloud server. Therefore, it is advisable to encrypt the data at each sensor node immediately after it is produced and incorporate secret sharing schemes to realise access control. In more details, each sensor node transmits the encrypted data to the gateway device by which data is integrated and encoded into shares of information. Due to security concerns, these shares are stored in separate cloud servers and the data retrieval must conform with the access policy. To realise this system, we present a novel research upon secret sharing in the encrypted domain.

Secret sharing is a study in cryptography originated independently by Blakley [6] and Shamir [7] in 1979. A secure (t, n)-threshold scheme is defined as splitting a secret message into n pieces of information in such a way that any fewer than $t \leq n$ pieces reveal no information about the secret. Only in the presence of t or more pieces will the secret be determined. Each piece of information is generally called a 'share' (as Sharmir's terminology) or a 'shadow' (as Blakley's terminology). An intuitive way of splitting a secret message, say, 'password' is to split it literally into shares: 'pa------', '--ss----', '----wo--', and '------rd'. This naïve approach is, however, insecure in the sense that every share leaks a part of the secret. Shamir proposed an elegant solution to share the secret in a secure manner. Suppose that a dealer wants to share a secret to n participants in such a way that only more than t participants pool their shares together will the secret be reconstructed. Let the secret be denoted

by s and we generate $t-1$ random numbers denoted by r_1, r_2, ..., and r_{t-1}. Then, we form a polynomial

$$f(x) \equiv s + r_1 x + r_2 x^2 + \cdots + r_{t-1} x^{t-1} \pmod{P}, \tag{1.1}$$

where P is a randomly chosen prime number. Let us draw any n points from the polynomial, for example, $(1, f(1))$, $(2, f(2))$, ..., and $(n, f(n))$, and distribute them to n participants respectively as shares. It is observed that there are t unknown variables in the polynomial and thus with t different points one is able to solve for the variables including the secret (*i.e.* the constant term). In other words, the reconstruction process is to simply use Lagrange interpolation to solve a set of t simultaneous equations. Shamir's scheme is algebraic in nature in contrast to Blakley's scheme based on geometric structures. As a toy example of Blakley's construction, consider the secret as a point in a three-dimensional space and the shadows as hyperplanes whose common intersection is the secret point. Any three of the planes suffice to identify the point. As each successive shadow is exposed, however, the range of possible values of the secret narrows. Since the introduction of secret sharing, numerous extended problems have appeared. The study towards a general access structure was considered by Ito, Saito, and Nishizeki [8] and had become a principal study since then [9–12]. To manage various malicious behaviour by dishonest parties, the notion of verifiable secret sharing was introduced by Chor *et al.* [13] and had been studied extensively thereafter [14–19]. Another closely related branch is visual cryptography originated by Naor and Shamir [20] for the secrecy of visual information, including greyscale, colour, and halftone images [21–23].

Over the past decades, secret sharing schemes have found various applications. One of the possible applications in the healthcare industry is to protect the privacy of patients' health records against cybersecurity threats while allowing efficient access for a group of authorised physicians and surgeons. In IoT-based healthcare applications, a patient's health record often contains medical data acquired from different sensor nodes. A full measure of privacy protection ought to even prevent data revelation between sensor nodes. More generally, we consider the problem of sharing multiple secrets (*i.e.* health data) generated from t different sources (*i.e.* sensor nodes) amongst a society of n participants (*i.e.* cloud servers of medical institutions). Every secret is prohibited from being revealed to another source as well as the dealer (*i.e.* gateway device). We remark that this statement can also be applied to other mutually distrustful situations, especially in the case of commercial applications. This problem, though different, is similar to that studied by Iirgemarsson and Simmons [24]. In their study, they noted that the problem of sharing the secret in the absence of a trusted dealer has been largely ignored by researchers in this area. In response to this, they introduced a two-level control protocol to share a secret determined by a democratic consent scheme without mutually trusted parties. Each participant equally contributes a private input to the determination of the secret and distributes the contribution among other participants through an autocratic sharing scheme.

In the following, let us consider two naïve solutions to our research problem and analyse their pros and cons. Among a variety of privacy protection mechanisms, encryption has a high level of reliability and universality. Naturally, the secrets are encrypted once they have been produced from the sources. The problem is therefore reduced to the sharing of encrypted data. Consider a key server who has a pair of public and private keys. The public key is used for encryption, whereas the private key is employed for decryption. The first solution is to create shares of the private key by arranging the key as the constant term in Eq. (1.1). The encrypted files, instead of being encoded as shares, are stored in a database. At the time when the number of collaborative participants are as many as required, the

private key will be reconstructed and then the files in the database can be deciphered. On the one hand, this solution is simple and the computational load of the sharing procedure is light. On the other hand, however, to access the secret files, one must perform one reconstruction algorithm for the key plus one decryption algorithm for the files. In addition to this, this scheme requires different pairs of public and private keys for different sets of secret files (*e.g.* different patients' health records); otherwise, once the participants reconstruct the private key, they will be able to decipher all the files stored in the database. Furthermore, storing all the important files in a central database may be vulnerable to a number of cyber attacks. Thus, it is reasonable to share the files to authorised participants to reduce the risk of cyber threats.

The second solution is that suppose there are t encrypted secrets denoted by $\mathcal{E}(s_0)$, $\mathcal{E}(s_1)$, ..., and $\mathcal{E}(s_{t-1})$. We form a polynomial by arranging t encrypted secrets as t coefficients in Eq. (1.1). More generally, we can assume that there are k encrypted secrets, where $k \leq t$, and choose $t - k$ random numbers as the rest of the coefficients to complete the polynomial. Either way, we can draw n points as the shares for individual participants. In the presence of t shares or more, the encrypted data will be reconstructed. With the decryption key, the data will eventually be revealed. In practice, this scheme has a non-trivial issue of key distribution amongst the participants. It may be addressed by one of the following approaches. First, use a secure channel to transmit the key to individual participants. Second, let the pair of encryption and decryption keys be generated by a key agreement protocol (*e.g.* Diffie–Hellman key exchange protocol [25]) amongst the group of participants, instead of being generated by the key server. Third, encrypt the key with each participant's public key and send it to the corresponding one as an instance of asymmetrical cryptography (*e.g.* elliptic curve cryptosystems [26]). Aside from the issue of key distribution, this scheme still requires extra efforts of participants, namely, one reconstruction step for the encrypted data plus one decryption step for the original data. It may be troublesome in particular situations. For instance, when there is a surgical emergency, the time delay for accessing health records becomes problematic. Hence, we conclude that these naïve solutions, though feasible, are deficient in several aspects, which motivate us towards finer constructions.

In this paper, we study how multiple sources are possible to share their secrets amongst a group of n participants without revealing their own secrets to one another. We analyse the pros and cons of several possible solutions and develop practical schemes: a simple (2, 2)-threshold scheme, an extended (n, n)-threshold scheme, and a generalised (t, n)-threshold scheme. The developed schemes follow Sharmir's construction in which a collusion of fewer than t participants has no better chance of guessing the secret than a non-participant who has no privileged information at all. The remainder of this paper is organised as follows. Section 2 gives the preliminaries of privacy homomorphisms. Section 3 reviews a naïve (2, 2)-threshold scheme. Section 4 discusses an extended (n, n)-threshold scheme. Section 5 studies a generalised (t, n)-threshold scheme. The paper is concluded in Section 6 with directions for future research.

2. Privacy homomorphisms

The term 'privacy homomorphisms' was coined by Rivest *et al.* to describe special encryption functions which permit encrypted data to be operated on [27]. These special algebraic mappings between the paintext and ciphertext spaces allow the result of operations on the ciphertexts, when deciphered, to match the result of operations on the plaintexts. Let us see a well-understood example of privacy

homomorphisms. Let p and q be two large primes and the modulus $N = p \cdot q$. Let e and d be the public and private keys of the RSA cryptosystem, respectively. Note that e and d satisfy the condition that

$$e \cdot d \equiv 1 \pmod{\phi(N)}, \tag{2.1}$$

where ϕ is Euler's phi function, *i.e.* $\phi(N) = (p-1)(q-1)$. The RSA cryptosystem has an encryption function

$$c \equiv m^e \pmod{N}, \tag{2.2}$$

and a decryption function

$$m \equiv c^d \pmod{N}, \tag{2.3}$$

where m denotes the message and c denotes the ciphertext. Suppose that we wish to generate the encrypted result which, when decrypted, matches the product of two messages m_1 and m_2 through the operations on the ciphertexts c_1 and c_2. This is achieved by

$$({m_1}^e) \cdot ({m_2}^e) \equiv (m_1 \cdot m_2)^e \pmod{N}. \tag{2.4}$$

For more information about the RSA cryptosystem, the reader is referred to [28].

Since the introduction of privacy homomorphisms, there has been a surge of interests in the design of homomorphic cryptosystems (*e.g.* ElGamal [29], Okamoto–Uchiyama [30], and Damgård–Jurik [31] cryptosystems). One of the well-studied homomorphic cryptosystems is the Paillier cryptosystem [32]. Let m_1 and m_2 be two arbitrary messages, N be the product of two large primes, $\mathcal{E}(\cdot)$ be the encryption function, and $\mathcal{D}(\cdot)$ be the decryption function. The Paillier cryptosystem permits homomorphic addition:

$$\mathcal{D}(\mathcal{E}(m_1) \cdot \mathcal{E}(m_2) \bmod N^2) \equiv m_1 + m_2 \pmod{N}, \tag{2.5}$$

and homomorphic multiplication:

$$\mathcal{D}(\mathcal{E}(m_1)^{m_2} \bmod N^2) \equiv m_1 \cdot m_2 \pmod{N}. \tag{2.6}$$

More details of the Paillier cryptosystem are described as follows. This consists of three phases: key generation phase, encryption phase, and decryption phase. In the key generation phase, we choose two large primes p and q. Then, we compute $N = pq$ and $\lambda = \text{lcm}(p-1, q-1)$, where 'lcm' stands for least common multiple. Afterwards, we select a random integer $g \in \mathbb{Z}/N^2\mathbb{Z}^*$ and calculate

$$\mu \equiv (L(g^\lambda \pmod{N^2}))^{-1} \pmod{N}, \tag{2.7}$$

where

$$L(x) = \frac{x-1}{N}. \tag{2.8}$$

As a result, the public key is (n, g) and the private key is (λ, μ). In the encryption phase, let m be a message to be encrypted and r be a randomly selected integer, where $m, r \in \mathbb{Z}/N\mathbb{Z}$. The ciphertext is then computed as

$$c \equiv g^m \cdot r^N \pmod{N^2}. \tag{2.9}$$

This scheme has a ciphertext expansion phenomenon as the message space is $\mathcal{M} = \mathbb{Z}/N\mathbb{Z}$ and the ciphertext spac is $C = \mathbb{Z}/N^2\mathbb{Z}^*$. In the decryption phase, the plaintext message is deciphered by

$$m \equiv L(c^{\lambda} \pmod{N^2}) \cdot \mu \pmod{N}. \tag{2.10}$$

It is observed that the decryption process involves a modular exponentiation, which is computationally expensive, with the addition of other operations of minor cost. Therefore, as aforementioned, in some time-sensitive applications, one would wish not to involve decryption in the secret reconstruction process. In the remainder of this paper, we assume all the homomorphisms applied are those of the Paillier cryptosystem unless otherwise specified. Nevertheless, the applicable homomorphisms are included but by no means limited to the homomorphisms of this particular cryptosystem.

3. (2, 2)-threshold secret sharing

Recently, we proposed a (2, 2)-threshold multi-secret sharing scheme to split a batch of two secrets into two shares via a semi-honest (or honest-but-curious) cloud service provider [33]. Only in the presence of two shares, the batch of two secrets can be restored. Let us describe how this scheme can solve the problem of privacy-preserving secret sharing. Let s_1 and s_2 be two secrets generated from two separate sources, respectively. To preserve the privacy of secrets, s_1 and s_2 are encrypted immediately after being produced. The encrypted secrets $\mathcal{E}(s_2)$ and $\mathcal{E}(s_2)$ are uploaded to the dealer for sharing. Let x_1 and x_2 be any integers that satisfy

$$\begin{aligned} &\gcd(x_1 + x_2, N) = 1, \\ &\gcd(x_1 - x_2, N) = 1. \end{aligned} \tag{3.1}$$

Note that 'gcd' stands for greatest common divisor. It is not difficult to find proper x_1 and x_2 because N is the product of two large primes. Since

$$x_1{}^2 - x_2{}^2 \equiv (x_1 + x_2) \cdot (x_1 - x_2) \pmod{N}, \tag{3.2}$$

we derive

$$\gcd(x_1{}^2 - x_2{}^2, N) = 1. \tag{3.3}$$

This also implies

$$\gcd(x_1{}^2 - x_2{}^2, N) = 1. \tag{3.4}$$

Then, two shares are created as

$$\begin{aligned} \mathcal{E}(y_1) &\equiv \mathcal{E}(s_1)^{x_1} \cdot \mathcal{E}(s_2)^{x_2} \pmod{N^2}, \\ \mathcal{E}(y_2) &\equiv \mathcal{E}(s_1)^{x_2} \cdot \mathcal{E}(s_2)^{x_1} \pmod{N^2}. \end{aligned} \tag{3.5}$$

Following the homomorphic properties, we rewrite

$$\begin{aligned} \mathcal{E}(y_1) &\equiv \mathcal{E}(s_1x_1 + s_2x_2) \pmod{N^2}, \\ \mathcal{E}(y_2) &\equiv \mathcal{E}(s_1x_2 + s_2x_1) \pmod{N^2}. \end{aligned} \tag{3.6}$$

The dealer distributes x_1 and x_2 to two participants and sends $\mathcal{E}(y_1)$ and $\mathcal{E}(y_2)$ to the key server for decryption. The decrypted results are

$$\begin{aligned} y_1 &\equiv s_1x_1 + s_2x_2 \pmod{N}, \\ y_2 &\equiv s_1x_2 + s_2x_1 \pmod{N}. \end{aligned} \tag{3.7}$$

Then, y_1 and y_2 are also dispensed to the participants. When the participants pool their shares (x_1, y_1) and (x_2, y_2) together, they compute

$$x_1y_1 - x_2y_2 \equiv ({x_1}^2 - {x_2}^2)s_1 \pmod{N}. \tag{3.8}$$

Note that

$$\begin{aligned} x_1y_1 &\equiv ({x_1}^2s_1 + x_1x_2s_2) \pmod{N} \\ x_2y_2 &\equiv ({x_2}^2s_1 + x_1x_2s_2) \pmod{N}. \end{aligned} \tag{3.9}$$

Since $\gcd({x_1}^2 - {x_2}^2, n) = 1$, we know there exists one and only one modular multiplicative inverse such that

$$({x_1}^2 - {x_2}^2) \cdot ({x_1}^2 - {x_2}^2)^{-1} \equiv 1 \pmod{N}. \tag{3.10}$$

The value of $({x_1}^2 - {x_2}^2)^{-1}$ can be solved by the extended Euclidean algorithm. Eventually, the secret s_1 is unveiled by

$$s_1 \equiv (x_1y_1 - x_2y_2) \cdot ({x_1}^2 - {x_2}^2)^{-1} \pmod{N}. \tag{3.11}$$

In the same manner, the secret s_2 is decoded as

$$s_2 \equiv (x_2y_1 - x_1y_2) \cdot ({x_2}^2 - {x_1}^2)^{-1} \pmod{N}. \tag{3.12}$$

It is worth noting that even though y_1 and y_2 have been disclosed to the key server during the process, s_1 and s_2 are still kept secret since the key server has no knowledge about x_1 and x_2. The secret reconstruction process does not involve the decryption operation and thus is time-efficient.

4. (n, n)-threshold secret sharing

Let us extend the previous (2, 2)-threshold scheme to a (n, n)-threshold scheme. For conciseness, we omit modulus symbols in the following description where there is no ambiguity. Let n secrets generated from sources be denoted by s_1, s_2, ..., and s_n. After encryption, the encrypted results, written as $\mathcal{E}(s_1)$, $\mathcal{E}(s_2)$, ..., and $\mathcal{E}(s_n)$, are transmitted to the dealer for sharing. The dealer chooses n random numbers x_1, x_2, ..., and x_n such that a matrix

$$\mathbf{X} = \begin{pmatrix} x_1 & x_2 & x_3 & \cdots & x_n \\ x_2 & x_3 & x_4 & \cdots & x_1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_n & x_1 & x_2 & \cdots & x_{n-1} \end{pmatrix} \tag{4.1}$$

has a modular multiplicative inverse $\mathbf{X}^{-1}$ in $\mathbb{Z}/N\mathbb{Z}$. Alternatively, $\mathbf{X}$ must satisfy $\gcd(\det(\mathbf{X}), N) = 1$ and $\det(\mathbf{X}) \neq 0$. Note that 'det' stands for determinant. Let the dealer compute

$$\begin{aligned} \mathcal{E}(y_1) &= \mathcal{E}(s_1)^{x_1}\mathcal{E}(s_2)^{x_2}\cdots\mathcal{E}(s_n)^{x_n}, \\ \mathcal{E}(y_2) &= \mathcal{E}(s_1)^{x_2}\mathcal{E}(s_2)^{x_3}\cdots\mathcal{E}(s_n)^{x_1}, \\ &\vdots \\ \mathcal{E}(y_n) &= \mathcal{E}(s_1)^{x_n}\mathcal{E}(s_2)^{x_1}\cdots\mathcal{E}(s_n)^{x_{n-1}}. \end{aligned} \tag{4.2}$$

According to the homomorphic properties, we derive

$$\begin{aligned} \mathcal{E}(y_1) &= \mathcal{E}(s_1x_1 + s_2x_2 + \cdots + s_nx_n), \\ \mathcal{E}(y_2) &= \mathcal{E}(s_1x_2 + s_2x_3 + \cdots + s_nx_1), \\ &\vdots \\ \mathcal{E}(y_n) &= \mathcal{E}(s_1x_n + s_2x_1 + \cdots + s_nx_{n-1}). \end{aligned} \tag{4.3}$$

The sharing process can be fulfilled by cloud computing to relieve the dealer of computational burdens without revealing the private information about the secrets. The dealer dispenses x_1, x_2, ..., and x_n to n participants respectively and passes $\mathcal{E}(y_1)$, $\mathcal{E}(y_2)$, ..., and $\mathcal{E}(y_n)$ to the key server for decryption. The decrypted results, written as

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \mathbf{X}\begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_n \end{pmatrix}, \tag{4.4}$$

are allocated to individual participants as well. When all the participants pool their shares together, they retrieve the secrets by

$$\begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_n \end{pmatrix} = \mathbf{X}^{-1}\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}. \tag{4.5}$$

Example. *Let us demonstrate that the previous (2, 2)-threshold scheme is actually a special case of the (n, n)-threshold scheme. In the case where there are two secrets s_1 and s_2 to be encoded, the dealer randomly chooses x_1 and x_2 to form a matrix*

$$\begin{pmatrix} x_1 & x_2 \\ x_2 & x_1 \end{pmatrix}.$$

Then, we compute

$$\begin{aligned} \mathcal{E}(y_1) &= \mathcal{E}(s_1)^{x_1}\mathcal{E}(s_2)^{x_2} = \mathcal{E}(s_1x_1 + s_2x_2), \\ \mathcal{E}(y_2) &= \mathcal{E}(s_1)^{x_1}\mathcal{E}(s_2)^{x_2} = \mathcal{E}(s_1x_2 + s_2x_1), \end{aligned}$$

which are equivalent to the results in Eq. (3.5) *and Eq.* (3.6)*. By decryption, we obtain*

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} x_1 & x_2 \\ x_2 & x_1 \end{pmatrix}\begin{pmatrix} s_1 \\ s_2 \end{pmatrix},$$

which are equal to the results in Eq. (3.7)*. Eventually, we retrieve the secrets by*

$$\begin{pmatrix} s_1 \\ s_2 \end{pmatrix} = \begin{pmatrix} x_1 & x_2 \\ x_2 & x_1 \end{pmatrix}^{-1} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix},$$

which are identical to the results in Eq. (3.11) *and Eq.* (3.12)*.*

As an extension of the (2, 2)-threshold scheme, this scheme has the same security strength. It is theoretically secure in the sense that any subset of participants has absolutely no knowledge about the secrets unless all the shares are in presence. The secret reconstruction procedure does not involve decryption. Thus, it is time-efficient and can be established without the means of key distribution.

5. (t, n)-threshold secret sharing

In light of the previous (n,n)-threshold scheme, we further derive a generalised (t, n)-threshold scheme. Before we proceed further, let us discuss some possible (t, n)-threshold schemes and analyse their pros and cons. Let $\{s_i\}_{i=1}^{t}$ denote t secrets generated from separate sources and P be a large prime. With Shamir's algorithm, the dealer constructs a polynomial

$$f(x) \equiv \sum_{i=1}^{t} s_i x^{i-1} \pmod{P}, \tag{5.1}$$

and draws n points $(x_1, f(x_1))$, $(x_2, f(x_2))$, $\ldots$, $(x_n, f(x_n))$ as shares for n participants. In our defined scenario, the secrets are encrypted into $\{\mathcal{E}(s_i)\}_{i=1}^{t}$ immediately after being produced. Let k denotes the decryption key. The first possible scheme is to split k into n shares by drawing n points from the following polynomial:

$$f_1(x) \equiv k + \sum_{j=1}^{t-1} r_j x^j \pmod{P}, \tag{5.2}$$

where $\{r_j\}_{j=1}^{t-1}$ are $t-1$ randomly chosen integers. The encrypted data has to be stored in a database so that when t or more participants reconstruct the key collaboratively, they can retrieve and decrypt the data. Nonetheless, the database may be vulnerable to numerous cyber attacks. The second possible scheme is to create shares according to the following polynomial:

$$f_2(x) \equiv \sum_{i=1}^{t} \mathcal{E}(s_i) x^{i-1} \pmod{P}. \tag{5.3}$$

When t or more participants co-operate, they can reconstruct the encrypted data. In order to decrypt the data, the scheme must engage a key distribution protocol to share the key amongst the participants. To compensate for the shortcomings, one may think of combining the previous two solutions and build a polynomial in the following form:

$$f_3(x) \equiv k + \sum_{j=1}^{t-1} \mathcal{E}(s_j) x^j \pmod{P}. \tag{5.4}$$

In this way, the authorised subset of participants is able to reconstruct and decrypt the data from the shares. With the knowledge of the key, however, the dealer is able to decipher the data and thus the privacy is threatened. Regrettably, as previous strategies all have obvious limitations, we need to find another way to do so.

For a moment, let us forget about the problem of sharing ciphertexts and only consider sharing the plaintexts since extending the idea to the sharing of the ciphertexts is easy once the following concepts are understood. Let $\mathbf{s}_{t,1}$ denote a vector of t secrets, $\mathbf{y}_{n,1}$ denote a vector of n shares, and $\mathbf{X}_{n,t}$ denote an $n \times t$ matrix. We define an encoding function

$$\mathbf{y}_{n,1} = \mathbf{X}_{n,t} \cdot \mathbf{s}_{t,1} \tag{5.5}$$

and a decoding function

$$\mathbf{s}_{t,1} = \mathbf{X}_{t,t}^{-1} \cdot \mathbf{y}_{t,1}, \tag{5.6}$$

where $\mathbf{y}_{t,1} \subset \mathbf{y}_{n,1}$, and $\mathbf{X}_{t,t} \subset \mathbf{X}_{n,t}$. In the case of (n, n)-threshold secret sharing, the above encoding and decoding functions are equivalent to Eq. (4.4) and Eq. (4.5), respectively. In the previous special case, we only require that $\mathbf{X}_{n,n}$ has a modular multiplicative inverse. In the current generalised case, however, we require that any $t \times t$ sub-matrix of $\mathbf{X}_{n,t}$ has a modular multiplicative inverse. In fact, when $t = n$, the current requirement reduces to the previous one since the one and only sub-matrix of $\mathbf{X}_{n,t}$ is $\mathbf{X}_{n,t}$ itself. Our question is hence 'is it possible to construct a valid matrix $\mathbf{X}_{n,t}$ such that any square matrix $\mathbf{X}_{t,t}$ consisting of t rows of $\mathbf{X}_{n,t}$ has a multiplicative inverse?'.

A matrix $\mathbf{A}$ is invertible if and only if its determinant is non-zero. When t and n are small numbers, we could use trial and error to construct a valid $\mathbf{X}_{n,t}$ such that $\det(\mathbf{X}_{t,t}) \neq 0$ for any $\mathbf{X}_{t,t}$. This approach is, however, not practical since collisions become difficult to be handled as the ratio between n and t, implying the number of possible combinations, grows large. To obtain a valid matrix in a systematic way, one of the possible solutions is to construct a Vandermonde matrix.

Definition (Vandermonde matrix)**.** *An $n \times t$ Vandermonde matrix has a form*

$$\mathbf{A}_{n,t} = \begin{pmatrix} {\alpha_1}^0 & {\alpha_1}^1 & {\alpha_1}^2 & \cdots & {\alpha_1}^{t-1} \\ {\alpha_2}^0 & {\alpha_2}^1 & {\alpha_2}^2 & \cdots & {\alpha_2}^{t-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ {\alpha_n}^0 & {\alpha_n}^1 & {\alpha_n}^2 & \cdots & {\alpha_n}^{t-1} \end{pmatrix}.$$

For a $t \times t$ square Vandermonde matrix, the determinant is given by

$$\det(\mathbf{A}_{t,t}) = \prod_{1 \leq i < j \leq t} (\alpha_j - \alpha_i).$$

Example. *Let us compute* $\det(\mathbf{A})$*, where*

$$\mathbf{A} = \begin{pmatrix} 1 & \alpha_1 & {\alpha_1}^2 \\ 1 & \alpha_2 & {\alpha_2}^2 \\ 1 & \alpha_3 & {\alpha_3}^2 \end{pmatrix}.$$

The determinant of $\mathbf{A}$ *is given by*

$$
\begin{aligned}
&\det\begin{pmatrix} 1 & \alpha_1 & {\alpha_1}^2 \\ 1 & \alpha_2 & {\alpha_2}^2 \\ 1 & \alpha_3 & {\alpha_3}^2 \end{pmatrix} \\
&= \det\begin{pmatrix} 1 & \alpha_1 & {\alpha_1}^2 \\ 0 & \alpha_2 - \alpha_1 & {\alpha_2}^2 - {\alpha_1}^2 \\ 0 & \alpha_3 - \alpha_1 & {\alpha_3}^2 - {\alpha_1}^2 \end{pmatrix} \\
&= \det\begin{pmatrix} \alpha_2 - \alpha_1 & {\alpha_2}^2 - {\alpha_1}^2 \\ \alpha_3 - \alpha_1 & {\alpha_3}^2 - {\alpha_1}^2 \end{pmatrix} \\
&= (\alpha_2 - \alpha_1)(\alpha_3 - \alpha_1)\det\begin{pmatrix} 1 & \alpha_2 - \alpha_1 \\ 1 & \alpha_3 - \alpha_1 \end{pmatrix} \\
&= (\alpha_2 - \alpha_1)(\alpha_3 - \alpha_1)\det\begin{pmatrix} 1 & \alpha_2 - \alpha_1 \\ 0 & \alpha_3 - \alpha_2 \end{pmatrix} \\
&= (\alpha_2 - \alpha_1)(\alpha_3 - \alpha_1)(\alpha_3 - \alpha_2)
\end{aligned}
$$

Corollary (Invertible Vandermonde matrix)**.** *A square Vandermonde matrix is invertible if and only if all α_i are distinct. When the condition suffices, the matrix has a nonzero determinant.*

Given the above preliminaries, we can start with the detailed construction of sharing ciphertexts. Consider a Vandermonde matrix written as

$$
\mathbf{X}_{n,t} = \begin{pmatrix} {x_1}^0 & {x_1}^1 & {x_1}^2 & \cdots & {x_1}^{t-1} \\ {x_2}^0 & {x_2}^1 & {x_2}^2 & \cdots & {x_2}^{t-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ {x_n}^0 & {x_n}^1 & {x_n}^2 & \cdots & {x_n}^{t-1} \end{pmatrix}, \tag{5.7}
$$

where $\{x_i\}_{i=1}^n$ are all distinct. The shares are created as

$$
\begin{aligned}
\mathcal{E}(y_1) &= \mathcal{E}(s_1)^{{x_1}^0}\mathcal{E}(s_2)^{{x_1}^1}\cdots\mathcal{E}(s_t)^{{x_1}^{t-1}}, \\
\mathcal{E}(y_2) &= \mathcal{E}(s_1)^{{x_2}^0}\mathcal{E}(s_2)^{{x_2}^1}\cdots\mathcal{E}(s_t)^{{x_2}^{t-1}}, \\
&\vdots \\
\mathcal{E}(y_n) &= \mathcal{E}(s_1)^{{x_n}^0}\mathcal{E}(s_2)^{{x_n}^1}\cdots\mathcal{E}(s_t)^{{x_n}^{t-1}}.
\end{aligned} \tag{5.8}
$$

Due to privacy homomorphisms, the above results are equivalent to

$$
\begin{aligned}
\mathcal{E}(y_1) &= \mathcal{E}(s_1{x_1}^0 + s_2{x_1}^1 + \cdots + s_t{x_1}^{t-1}), \\
\mathcal{E}(y_2) &= \mathcal{E}(s_1{x_2}^0 + s_2{x_2}^1 + \cdots + s_t{x_2}^{t-1}), \\
&\vdots \\
\mathcal{E}(y_n) &= \mathcal{E}(s_1{x_n}^0 + s_2{x_n}^1 + \cdots + s_t{x_n}^{t-1}).
\end{aligned} \tag{5.9}
$$

After decryption, the results become

$$\begin{aligned} y_1 &= s_1 {x_1}^0 + s_2 {x_1}^1 + \cdots + s_t {x_1}^{t-1}, \\ y_2 &= s_1 {x_2}^0 + s_2 {x_2}^1 + \cdots + s_t {x_2}^{t-1}, \\ &\vdots \\ y_n &= s_1 {x_n}^0 + s_2 {x_n}^1 + \cdots + s_t {x_n}^{t-1}, \end{aligned} \tag{5.10}$$

or alternatively, as expressed in Eq. (5.5). Each participant will receive a share (x_i, y_i), where $i \in \{1, 2, \ldots, n\}$. Suppose that a subset of participants has gathered a collection of shares, say, (x_j, y_j), where $j \in \{1, 2, \ldots, t\}$. Hence, the participants form a matrix

$$\mathbf{X}_{t,t} = \begin{pmatrix} {x_1}^0 & {x_1}^1 & {x_1}^2 & \cdots & {x_1}^{t-1} \\ {x_2}^0 & {x_2}^1 & {x_2}^2 & \cdots & {x_2}^{t-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ {x_t}^0 & {x_t}^1 & {x_t}^2 & \cdots & {x_t}^{t-1} \end{pmatrix}, \tag{5.11}$$

and reconstruct the secrets with Eq. (5.6). Note that $\mathbf{X}_{t,t}$ is a square Vandermonde matrix, thus invertible. The reader may have observed that when $\mathbf{X}_{n,t}$ is a Vandermonde matrix, Eq. (5.5) and Eq. (5.6) are the encoding and decoding functions of Shamir's scheme *per se*. Let $f(x)$ denote Shamir's encoding function, while $g(x)$ denote ours. The connection between two functions can be expressed as

$$g(x) = \prod_{i=1}^{t} \mathcal{E}(s_i)^{x^{i-1}} = \mathcal{E}(\sum_{i=1}^{t} s_i x^{i-1}) = \mathcal{E}(f(x)). \tag{5.12}$$

Except for the processing domain (either the plaintext or ciphertext domain), a noticeable difference between two schemes is the decoding process for which Shamir uses the Lagrange interpolation and we utilise a matrix multiplication. We remark that there are many studies on fast algorithms for matrix inversion [34] and multiplication [35–37].

6. Conclusion

In this paper, we address a novel research problem of secret sharing in the encrypted domain for IoT-based healthcare applications. We study the problem of sharing encrypted data, acquired from different sensor nodes, among a set of cloud servers. In conclusion, the proposed schemes are theoretically secure in the following senses. First, since the secret data is concealed by a secure encryption algorithm immediately after its creation, the dealer as well as other sources cannot access the secret data. Second, the key server only has partial shares and thus is also unable to retrieve the secret data. Third, conforming with the access policy, a subset of fewer than a certain number of participants does not suffice to decode the secrets either. In addition to this, the data is not required to be stored in a common database so that the scheme is not vulnerable to cyber threats against the database. Furthermore, since data retrieval does not involve computationally expensive decryption operations, the scheme is advantageous in time-sensitive circumstances. In the near future we intend to extend this work into a more general access structure based on the assumption that there are dishonest parties involved. Another line of further investigation is the application of this work in visual cryptography.

Acknowledgments

This work was supported by the Marie Skłodowska-Curie actions of EU Horizon 2020 programme through the project entitled 'Computer Vision Enabled Multimedia Forensics and People Identification' (Project No. 690907, Acronym: IDENTITY).

Conflict of interest

All authors declare no conflicts of interest in this paper.

References

1. S. Sharma, K. Chen and A. Sheth, Toward practical privacy-preserving analytics for IoT and cloud-based healthcare systems, *IEEE Int. Comput.*, **22** (2018), 42–51.
2. M. Elhoseny, G. Ramírez-González, O. M. Abu-Elnasr, et al., Secure medical data transmission model for IoT-based healthcare systems, *IEEE Access*, **6** (2018), 20596–20608.
3. T. Wu, F. Wu, J. M. Redouté, et al., An autonomous wireless body area network implementation towards IoT connected healthcare applications, *IEEE Access*, **5** (2017), 11413–11422.
4. F. Sebbak and F. Benhammadi, Majority-consensus fusion approach for elderly IoT-based healthcare applications, *Ann. Telecommun.*, **72** (2017), 157–171.
5. U. Satija, B. Ramkumar and M. S. Manikandan, Real-time signal quality-aware ECG telemetry system for IoT-based health care monitoring, *IEEE Internet Things J.*, **4** (2017), 815–823.
6. G. R. Blakley, Safeguarding cryptographic keys, in *Proc. AFIPS Nat. Comput. Conf. (NCC)*, New York, NY, USA, (1979), 313–317.
7. A. Shamir, How to share a secret, *Commun. ACM*, **22** (1979), 612–613.
8. M. Ito, A. Saito and T. Nishizeki, Secret sharing scheme realizing general access structure, in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Tokyo, Japan, (1987), 99–102.
9. J. Benaloh and J. Leichter, Generalized secret sharing and monotone functions, in *Proc. Conf. Theory and Appl. of Cryptography (CRYPTO)*, Santa Barbara, CA, USA, (1988), 27–35.
10. E. F. Brickell, Some ideal secret sharing schemes, in *Proc. Workshop Theory and Appl. of Cryptographic Techn. (EUROCRYPT)*, Houthalen, Belgium, (1989), 468–475.
11. E. F. Brickell and D. M. Davenport, On the classification of ideal secret sharing schemes, *J. Cryptology*, **4** (1991), 123–134.
12. A. Beimel and B. Chor., Universally ideal secret-sharing schemes, *IEEE Trans. Inf. Theory*, **40** (1994), 786–794.
13. B. Chor, S. Goldwasser and S. Micali, et al., Verifiable secret sharing and achieving simultaneity in the presence of faults, in *Proc. Ann. Symp. Found. Comput. Sci. (SFCS)*, Portland, OR, USA, (1985), 383–395.
14. P. Feldman, A Practical scheme for non-interactive verifiable secret sharing, in *Proc. Ann. Symp. Found. Comput. Sci. (SFCS)*, Los Angeles, CA, USA, (1987), 427–438.

15. T. Rabin and M. Ben-Or, Verifiable secret sharing and multiparty protocols with honest majority, in *Proc. Ann. ACM Symp. Theory of Comput. (STOC)*, Seattle, WA, USA, (1989), 73–85.

16. M. Tompa and H. Woll, How to share a secret with cheaters, *J. Cryptol.*, **1** (1989), 133–138.

17. T. P. Pedersen, Non-interactive and information-theoretic secure verifiable secret sharing, in *Proc. Annl. Int. Cryptology Cof. (CRYPTO)*, Santa Barbara, CA, USA, (1991), 129–140.

18. M. Stadler, Publicly verifiable secret sharing, in *Proc. Int. Conf. Theory and Appl. of Cryptographic Techn. (EUROCRYPT)*, Saragossa, Spain, (1996), 190–199.

19. R. Cramer, I. Damgård and U. Maurer, General secure multi-party computation from any linear secret-sharing scheme, in *Proc. Int. Conf. Theory and Appl. of Cryptographic Techn. (EUROCRYPT)*, Bruges, Belgium, (2000), 316–334.

20. M. Naor and A. Shamir, Visual cryptography, in *Proc. Workshop Theory and Appl. of Cryptographic Techn. (EUROCRYPT)*, Perugia, Italy, (1994), 1–12.

21. C. Blundo, A. D. Santis and M. Naor, Visual cryptography for grey level images, *Inf. Process. Lett.*, **75** (2000), 255–259.

22. Y. C. Hou, Visual cryptography for color images, *Pattern Recognit.*, **36** (2003), 1619–1629.

23. Z. Zhou, G. R. Arce and G. D. Crescenzo, Halftone visual cryptography, *IEEE Trans. Image Process*, **15** (2006), 2441–2453.

24. I. Ingemarsson and G. J. Simmons, A protocol to set up shared secret schemes without the assistance of a mutually trusted party, in *Proc. Workshop Theory and Appl. of Cryptographic Techn. (EUROCRYPT)*, Aarhus, Denmark, (1994), 266–282.

25. W. Diffie and M. Hellman, New directions in cryptography, *IEEE Trans. Inf. Theory*, **22** (1976), 644–654.

26. N. Koblitz, Elliptic curve cryptosystems, *Math. Comput.*, **48** (1987), 203–209.

27. R. L. Rivest, L. Adleman and M. L. Dertouzos, On data banks and privacy homomorphisms, in *Foundations of Secure Computation* (eds. R. J. Lipton, D. P. Dobkin, and A. K. Jones), Academic Press, (1978), 169–180.

28. R. L. Rivest, A. Shamir and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, *Commun. ACM*, **21** (1978), 120–126.

29. T. Elgamal, A public key cryptosystem and a signature scheme based on discrete logarithms, *IEEE Trans. Inf. Theory*, **4** (1985), 469–472.

30. T. Okamoto and S. Uchiyama, A new public-key cryptosystem as secure as factoring, in *Proc. Int. Conf. Theory and Appl. of Cryptographic Techn. (EUROCRYPT)*, Espoo, Finland, (1998), 308–318.

31. I. Damgård and M. Jurik, A generalisation, a simplication and some applications of Paillier's probabilistic public-key system, in *Proc. Int. Workshop Practice and Theory in Public Key Cryptography (PKC)*, Cheju Island, Korea, (2001), 119–136.

32. P. Paillier, Public-key cryptosystems based on composite degree residuosity slasses, in *Proc. Int. Conf. Theory and Appl. of Cryptographic Techn. (EUROCRYPT)*, Prague, Czech Republic, (1999), 223–238.

33. C. C. Chang and C. T. Li, Secure secret sharing in the cloud, in *Proc. IEEE Int. Symp. Multimedia (ISM)*, Taichung, Taiwan, (2017), 358–361.
34. L. Csanky, Fast parallel matrix inversion algorithms, *SIAM J. Comput.*, **5** (1976), 618–623.
35. V. Strassen, Gaussian elimination is not optimal, *Numerische Mathematik*, **13** (1969), 354–356.
36. D. Coppersmith and S. Winograd, Matrix multiplication via arithmetic progressions, *J. Symbolic Comput.*, **9** (1990), 251–280.
37. F. Le Gall, Powers of tensors and fast matrix multiplication, in *Proc. Int. Symp. Symbolic and Algebraic Comput. (ISSAC)*, Kobe, Japan, (2014), 296–303.

AIMS Press