Cloud-Assisted Image Double Protection System With Encryption and Data Hiding Based on Compressive Sensing

Di Xiao, Chongqing University, China*
Jia Liang, Chongqing University, China
Yanping Xiang, Chongqing University, China
Jiaqi Zhou, Chongqing University, China

ABSTRACT

In this paper, the authors propose a compressive sensing (CS)-based scheme that combines encryption and data hiding to provide double protection to the image data in the cloud outsourcing. Different domain techniques are integrated for efficiency and security. After the data holder gets the sample of the raw data, he embeds watermark into sample and encrypts it, and then sends the protected sample to cloud for storage and recovery. Cloud cannot get any information about either the original data or the watermark in the CS recovery process. Finally, users can extract the watermark and decrypt the data recovered directly in sparse domains. At the same time, after extracting the watermark, the image data of the user will be closer to the original data compared with the data without extraction. In addition, the counter (CTR) mode is introduced to generate the measurement matrix of CS, which can improve security while avoiding the storage of measurement matrixes. The experimental results demonstrate that the scheme can provide both privacy protection and copyright protection with high efficiency.

KEYWORDS
Cloud-Assisted, Compressive Sensing, Data Hiding, Image Encryption, Privacy-Protection

INTRODUCTION

With the development of ubiquitous computing, many small, inexpensive, networked processing devices, such as wireless sensor and mobile terminal, have become more and more popular and performed a critical function in environmental protection, transportation, health and other fields. For example, in a wireless sensor network (Estrin, Govindan, Heidemann, & Kumar, 1999), sensor nodes usually function as an information collector to sense, capture, process and monitor a variety of monitoring information, and provide a wealth of detailed data. Nevertheless, since their computational capability and information processing capability are poor, and they are vulnerable to the problem of insufficient processing power. The contradictions are even worse for multimedia data, such as image.
How to cope with a huge amount of data when using resource-constrained devices becomes a primary issue for us to consider. Compressed sensing (CS) (Donoho, 2006) can sample the signal at a much lower sampling frequency than Nyquist theory. The CS sampling process of signal $x, x \in \mathbb{R}^n$, is done by matrix multiplication, i.e. $y = \Phi x$, where $\Phi$ is a measurement matrix with the size of $m \times n$ and $y \in \mathbb{R}^m$ is the measurement vector and $m << n$. CS compresses data while sampling to reduce transmission bandwidth consumption and save energy, which provides a new idea for solving the problem of resource limitation of information acquisition device. On the other hand, CS can be regarded as a symmetric encryption system. When using CS for sample compression, protecting confidentiality can be achieved without additional cost or minimal cost. The application of CS can greatly improve the efficiency of sensor information collection and transmission. However, since the process of CS reconstruction is relatively complicated (Karmarkar, 1984), many related studies cannot be directly applied to the situation where the receiving end resource is limited.

Cloud is a hosting technology for computing, storing, processing and sharing data. Cloud has huge storage space and strong information processing capability. In order to store and manage large amounts of image data more efficiently, a good choice is to outsource the storage and processing of these data to cloud. In many cloud applications, the data that the user needs to use is not uploaded by the user himself, but is provided by the sampling end or other users, namely the data holder. In this case, the data in cloud will be uploaded by the data holder, and be downloaded and used by the legal user only.

By combining CS with cloud, CS is utilized to collect data while the storage and calculation of data are managed by the cloud. It can not only fully reflect the advantages of both compressed sensing and cloud, but also make up for the shortcomings of each other. And thus, the burden of data holder and user can be greatly reduced. However, since the cloud is an open platform, how to ensure the security of data uploaded to the cloud will be a crucial issue. At the same time, since digital products are particularly vulnerable to plagiarism and illegal copying during transmission and exchange, copyright protection is also desirable.

Transmission and storage of big data require a lot of resources, but many information collection devices have limited resources. In order to prevent eavesdropping and other attacks, the data needs to be encrypted before transmission and storage. Orsdemir et al. (Orsdemir, Altun, Sharma, & Bocko, 2008) theoretically proved that the use of CS technology to encrypt has a certain robustness and security. Rachlin et al. (Rachlin & Baron, 2008) pointed out that CS can provide computational security, although compression perception does not reach the definition of Shannon’s perfect security. Therefore, CS can perform compression and encryption while sampling. In (D. Wu, Yang, Wang, & Wang, 2016), Distributed Compression Sensing (DCS) is used to compress and measure the original multimedia sensor data and upload the compressed data measurement results to the receiver. But the measurement matrix is used as a key, which requires a huge amount of storage space. In (Peng, Tian, & Kurths, 2017), CS of Semiconductor Products (STP-CS) is proposed. By breaking through the dimension matching limitation of matrix multiplication, the size of the key is reduced to save storage space, while the computational load and computing resources are also reduced. In order to further enhance the security, homomorphic encryption function is used to prevent traffic analysis and traffic tracing and to achieve privacy protection in (Xie et al., 2016). However, since the data reconstruction has been done at the receiving end, the aforementioned works are not suitable for the receiving end with limited resources.

A series of studies related to our work are to securely outsource computing to public cloud for storage and computation. To date, many protocols have been designed for secure outsourcing. Wang et al. (Wang, Zhang, Ren, & Roveda, 2014) proposed a cloud-assisted security monitoring system based on CS. The linear programming (Wang, Ren, & Wang, 2011) method is adopted to provide good protection for the input data in the cloud outsourcing service. At the same time, any information related to the original information cannot be obtained from the data output after security calculation. However, their solution brings the data holder a huge additional computational load. In order to reduce
the burden and improve the efficiency, Wu et al. (X. Wu, Tang, & Yang, 2014) replaced the linear programming by adopting the permutation matrix as the encryption key, and also provided good protection for the data in the cloud outsourcing service. The overload they generate is less, but the security is lower than that of Wang et al. In (Huang, Xu, Fu, & Wang, 2017), a collaborative OMP protocol is designed to outsource compute-intensive image restoration to the cloud while preserving the privacy of the image, but it requires two cloud servers to assist in the reconstruction.

It is generally believed that the digital watermarking technology is the most effective way to protect the copyright of the data. Based on the extensive development of digital watermarking technology, digital watermarking based on CS has also got some preliminary research results (Pan, Li, Yang, & Yan, 2015; Valenzise, Tagliasacchi, Tubaro, Cancelli, & Barni, 2009; Xiao, Cai, Wang, & Bai, 2016). Most of the current schemes implement information hiding by modifying CS measurements. They require data reconstruction before the watermark embedding process, which greatly increases the energy consumption of the sender and makes the measurements unable to be directly transmitted. Therefore, these schemes cannot be applied to large-scale data transmission of resource-constrained devices. Therefore, in this paper, we propose a cloud-assisted image double protection system based on CS which is suitable for both data holder and user with limited resources. In this system, CS is used for data collection and watermark embedding, and the counter (CTR) mode is introduced into the measurement matrix generation to improve security while avoiding the storage of measurement matrixes at sender and receiver; Reconstruction is outsourced to cloud while preserving data privacy; The embedded watermark is used to protect data copyright, and it does not need to reconstruct data when embedding.

The main contributions include:

- The proposed scheme achieves both encryption and data hiding at the same time, which is based on outsourcing CS reconstruction to the cloud.
- Due to the features of the CS, it adopts various techniques of different domains, which achieves the compression and data hiding in the measure value. Then the user can decode the image and extract the watermark in the frequency domain after the CS reconstruction.
- The counter (CTR) mode is introduced to generate the measurement matrix of CS in this scheme, which can effectively improve security while avoiding the storage of measurement matrixes.
- Although the original data cannot be fully recovered based on CS, the user can get more accurate data after extracting the watermark compared to the data without extracting the watermark.

**PROPOSED SCHEME**

In this paper, we propose a cloud-assisted double protection scheme based on CS which achieves both encryption and data hiding. Encryption can protect the privacy of raw data, while the watermark can be used for other purposes, such as copyright verification of raw data. The diagram of the proposed scheme is given in Figure 1. There are three parties: the data holder (DH) of the raw image data, the data user with constrained resources, and the cloud. Firstly, data holder gets the sample y of sparse signal f through CS, and then data holder could embed some secret data (such as the ID of data holder or image identification, we will use ID as the secret data in this paper) into y with our proposed watermark pattern. For privacy protection, data holder will not outsource y directly. Instead, he outsources an encrypted version y* of y and some associated metadata to cloud. Next, if user wants to use the image, he will send a request to cloud, then cloud directly reconstructs an output α* over the encrypted y* and sends α* to user. When obtaining α*, user will first decrypt it to α with the shared secret key K between data holder and user, then extract the secret data from α and recover the original image simultaneously. Based on CS, the scheme can achieve double protection for the image data.
As shown in Figure 1, the work of cloud is to recover the original data with any CS re-construction algorithm. Therefore, in the following, we focus on the other two parties to describe the scheme details: Data holder and User.

**Data Holder**

**Compressive Sensing**

Given an $k$-sparse signal $f$ with the size of $n \times 1$ ($k$-sparse means that there are at most $k$ non-sparse values in signal $f$). If we left-multiply a selected $m \times n$ ($m \ll n$) matrix $\Phi$, which is called measurement matrix to $f$, we can get a $m \times 1$ sample vector through $y = \Phi f$. In the sampling process, one fact is that real world data $f$ might not be always sparse. But as long as it can be represented as a $n\times 1$ sparse vector $\alpha$ under some properly chosen sparse basis $\varphi \in \mathbb{R}^{n \times n}$ via $f = \varphi \alpha$, we can still use CS theory and have $y = \Phi f = \Phi \varphi \alpha$.

This sampling process can also be regarded as an encryption process, where the measurement matrix is the key. So the measurements can be directly stored in cloud. However, using the same measurement matrix to encrypt multiple signals helps the attacker learn useful information to get the key. In order to solve this problem, the counter (CTR) mode is used to generate the measurement matrix in this paper, which can resist the chosen plaintext attack (CPA). The Gaussian random matrix is used to compress and encrypt the i-th signal. The measurement matrix $\Phi_i$ is constructed as follows:

**Step 1:** Given an initial vector IV with a length of N-bit, then:

$$CTR_i = IV, CTR_i = (CTR_{i-1} + 1) \mod 2^n$$  \hspace{1cm} (1)

**Step 2:** A hash function is chosen to generate 0-1 binary seeds to produce random numbers, as shown in (2), where $\parallel$ means that the data is placed after the previous data, and $K_i$ is the key used to generate the measurement matrix, which is owned by data holder:

$$seed_i = H\left(K_i \parallel CTR_i \parallel j\right), j \in (1, m)$$ \hspace{1cm} (2)

---

**Figure 1. Diagram of the proposed scheme**

---

---
**Step 3:** The pseudorandom numbers obeying Gauss distribution are generated under the control of the seeds, and then the measurement matrix is generated, as shown in (3):

\[
\Phi_i = \begin{bmatrix}
\Phi_i^1 \\
\cdots \\
\Phi_i^m
\end{bmatrix}, \text{ where } \Phi_i^j = \text{Random}(\text{seed}_i^j)
\]  

(3)

where \text{Random}(\cdot) represents a pseudorandom number generator. If the length of the binary string produced by the hash function is longer than the length of the seed L, then the first L bits are taken. In addition, the data holder needs to securely share the key \( K_1 \) and initial vector with the cloud server, and then the cloud server can generate the measurement matrix in the same way.

Here, let \( A = \Phi_i \varphi \). If matrix \( A \) satisfies Restricted Isometry Property (RIP) (Candes & Romberg, 2006), then the sparse \( f \) could be recovered with high probability from \( y \) by solving an \( l_1 \)-minimization problem (Candes & Tao, 2006):

\[
\min \| \alpha \|_1 \text{ subject to } y = A\alpha
\]  

(4)

**Watermark Pretreatment**

Along the flow chart, data holder first embeds the secret watermark \( \omega \) into the sample \( y \), but data holder does not embed the original watermark with value ‘1’ and ‘-1’. Data holder needs to perform watermark pretreatment to make the watermark format meet the requirement of subsequent watermark embedding.

From the flow chart, we can see that data holder should embed the watermark into \( y \) and the user will extract it from \( \alpha' \). We notice the significant information that data holder gets the sample \( y \) from some sensors directly, so all the items data holder have include the sample \( y \), the measurement matrix \( \Phi_i \) and the sparse basis \( \varphi \). Data holder does not know the value of \( \alpha \) which is reconstructed by CS, but data holder can know \( \alpha \) is sparse after reconstruction, and the approximate positions of both the non-sparse values and the sparse values in vector \( \alpha \) for some sparse basis such as DCT, DWT. Only sparse data can be reconstructed by CS, so in our scheme, the original watermark \( W \) should be transformed into similar sparse data \( \omega \) for subsequent embedding. According to our scheme and the property of CS, we introduce this watermark pretreatment algorithm as follows: (we need two parameter values, \( t \) and \( n \), to control the embedding content and accuracy of extracting, and hold the sparsity of the reconstructed data by cloud, where \( t \) denotes the scaling factor, a positive number \( d \) is the interval of binary bit, and \( n \) is equal to the length of vector \( \alpha \)).

Let the original watermark \( W \) take the values from \( \pm 1 \), then the \( \omega \) (initialized as an all-zero vector) is obtained via \( \omega(n - 1 - d * i) = t * W(i) \), where \( i \) is the sample index. For example: if the watermark \( W = \{1, -1, 1\}, n = 12, t = 5, d = 2 \), then we have \( \omega = \{0, 0, 0, 0, 0, 0, 5, 0, -5, 0, 5\} \). After watermark pretreatment, we transform the original watermark to a \( n \times 1 \) vector for watermark embedding in (5). The reason why we choose the back position of \( \omega \) to perform watermark pretreatment is that there are a large number of sparse values in \( \alpha \). For instance, the majority back position elements of a nature signal after DCT are high frequency data which are near to zero. If we embed the watermark information in these locations, the watermark can be recovered by CS.
**Watermark Embedding**

After watermark pretreatment, the original watermark has been changed into the format the next step needs, and then data holder can embed the watermark into the sample $y$:

$$y + A\omega = A(\alpha + \omega)$$  \hspace{1cm} (5)

In this step, let the measurement with watermark be $y' = y + A\omega$ and the recovered data with watermark be $\alpha' = \alpha + \omega$, then:

$$y' = A\alpha'$$  \hspace{1cm} (6)

After this step, $\alpha'$ is $k_1$-sparse, where $k_1 = k + "the\ length\ of\ W"$. Because the natural image can meet the sparseness requirements of CS, we are able to control the amount of the watermark to make $k_1$ meet the requirements of CS (Huang et al., 2017) so that the data and the watermark can be well recovered. Since (7) has the same form as (4), we have a new $l_1$-min problem:

$$\min \|\alpha'\| \text{ subject to } y' = A\alpha'$$  \hspace{1cm} (7)

In this way, data hiding has been done. In this case, UR can get $\alpha'$ through CS reconstruction, but the cloud can get most information about raw data by solving (7) so that the security of raw data cannot be guaranteed. In the next, we will introduce our encryption method based on (7).

**Data Encryption**

We left-multiply a random $m \times m$ matrix $M$ to (6) to encrypt both the sample $y'$:

$$My' = MA\alpha'$$  \hspace{1cm} (8)

Then we left-multiply a random $n \times n$ permutation matrix $Q$ to $\alpha'$ in order to prevent cloud from obtaining some information from the reconstructed data. Let $\alpha^* = Q\alpha'$, then we get:

$$My' = MAQ^{-1}\alpha^*$$  \hspace{1cm} (9)

Let $y^* = My'$ and $A^* = MAQ^{-1}$, then we have:

$$y^* = A^*\alpha^*$$  \hspace{1cm} (10)

An $n \times n$ $(0,1)$ matrix $Q$ has only one ‘1’ in every row and only one ‘1’ in every column, which can be constructed by PRNG, and the steps are as follows:

**Step1:** Initialize $Q$ to be a matrix of all zeros.

**Step2:** Use PRNG to generate a pseudo-random sequence $r$ under the control of key $K_2$. 


**Step3:** Sort \( r \) in ascending order to get \( r_i \), and then find the index \( v \) of every element of \( r_i \) in \( r \).

Finally, \( Q^{(i)}_j = 1 \), where \( Q^{(i)}_j \) is the element of \( Q \) in row \( i \) and column \( v(i) \).

For instance, \( r = \{0.25, 0.12, 0.01\} \), \( r_i = \{0.01, 0.12, 0.25\} \), \( v = \{3, 2, 1\} \):

\[
Q_i^3 = 1, Q_i^2 = 1, Q_i^1 = 1, Q = \begin{bmatrix}
0 & 0 & 1 \\
0 & 1 & 0 \\
1 & 0 & 0
\end{bmatrix}
\]

We can find that (10) has the same form as (4). If (10) satisfies the constraints of CS, (10) can be solved by cloud effectively. The constraints are that the matrix \( A^* \) must satisfy RIP and \( \alpha^* \) must be a sparse data. For the first condition \( A^* = MAQ^{-1} \), we can generate \( M \) by sampling entries from the normal distribution to make MA satisfy RIP. Further, for the permutation matrix \( Q \), since the location within the matrix element does not affect RIP, the matrix \( A^* \) satisfies RIP. At the same time, we have already known that \( \alpha' \) is \( k_1 \)-sparse and \( Q \) only affects the element positions, so \( \alpha^* = Q\alpha' \) is also \( k_1 \)-sparse.

In summary, in the above transformation, we use the encryption key \( K \) to generate \( M \) by CTR mode described in section of Compressive Sensing, where the matrix \( M \) contains \( m \times m \) random entries and the matrix \( Q \) contains \( n! \) different forms. Then we have a new \( l_1 \)-min problem as follows:

\[
\min \|\alpha^*\| \quad \text{subject to} \quad y^* = A^*\alpha^* \tag{11}
\]

Based on the above discussion, (11) can be solved by CS theory. In the following, we summarize the procedures of data holder:

**Step 1:** Data holder generates measurements matrix \( \Phi_i \) according to the key \( K_1 \). Let \( A = \Phi_i\varphi \).

**Step 2:** Data holder gets \( y \) from \( f \) directly through \( y = Af \).

**Step 3:** Data holder pretreats the watermark \( W \), and then embeds \( \omega \) into \( y \) to get \( y' \) through \( y' = y + A\omega \).

**Step 4:** Data holder encrypts \( y' \) to \( y^* \) and calculates \( A^* (y^* = My^' \text{ and } A^* = MAQ^{-1}) \), and then uploads \( y^* \) and \( A^* \) to cloud for storage and recovery. At the same time, data holder sends the secret key to user through a safe channel.

**User**

Considering users are usually resource-constrained devices like smart phones, so we shift the main computation, i.e. CS reconstruction algorithm, to cloud. The data reconstructed by cloud is \( \alpha^* \), so the first job of user is to decrypt \( \alpha^* \). Since \( \alpha^* = Q\alpha' \), user just needs to generate \( Q \) using key \( K_2 \) and do simple matrix multiplication computation \( \alpha^1 = Q^{-1}\alpha^* \).

After \( \alpha^1 \) is reconstructed, watermark information is also reconstructed and contained in \( \alpha' \). The following job of user is to extract the watermark \( W \). For the sparse vector \( \alpha \), the significant low frequency information concentrates on the first half part, while most of the second half part are zero. For \( \omega \), which is obtained by watermark pretreatment, its significant information is mainly located at the second half part, while the other part is zero. Therefore, for the reconstructed \( \alpha' \), there are
two locations with significant information: one is at the front end; the other is at the backmost end. For user, it is possible for him to obtain the watermark by only processing the back significant data. As described above, the position of the watermark \( W \) in \( \alpha' \) is decided by the corresponding position in \( \omega \). So user can extract watermark \( W \) from \( \alpha' \) by judging the positive and negative numbers in the corresponding position (if \( \alpha' \left( n - 1 - d \times i \right) \) is greater than 0, then \( W \left( i \right) \) takes “1”, otherwise \( W \left( i \right) \) takes “-1”, where \( i \) is the sample index). After that, user can get watermark vector \( \omega \) through watermark pretreatment again. By subtracting \( \omega \) from \( \alpha' \), user can get \( \alpha \), which is closer to the original signal compared with \( \alpha' \) for \( \alpha = \alpha' - \omega \). Finally, user recovers the image \( f = \varphi \alpha \).

In the following, we also summarize the procedures of user:

**Step 1:** Once one user wants to use the data stored in cloud, he will send request to cloud, cloud will reconstruct data \( \alpha^* \) by (11), and then convey \( \alpha^* \) to user.

**Step 2:** Since \( \alpha^* \) is encrypted, user will first decrypt it to \( \alpha' \), then extract secret watermark \( W \) included in \( \alpha' \), and verify the watermark. At the same time, user can get the original data \( \alpha \) with higher quality.

**Step 3:** Finally, user recovers the image \( f = \varphi \alpha \).

**EXPERIMENT AND ANALYSIS**

For experimental evaluation, MATLAB and MOSEK optimization toolbox are used on the workstation with an Intel Core i3 running at 3.30 GHz and 2GB RAM. DCT basis is considered as the representation basis. OMP algorithm is used to solve \( l_1 \) optimization problem. In the experiment, the grayscale images in the USC-SIPI image database with a size of 256×256 were used. For efficiently implementing experiment, the image is divided into smaller non-overlap image blocks with the same size of 32×32, and each block will be transformed to a 1024×1 vector via row sequence for CS.

**Security**

The process of watermark embedding is controlled by \( A = \Phi \varphi \), where the measurements matrix is generated by data holder according to the key \( K_i \). By embedding the watermark into the sample \( y \), \( y' = y + A \omega \). Due to the specificity of the embedding method, it is hard to get any information about the watermark \( \omega \) from \( y' \).

We know \( \alpha' = Q \alpha' \) and \( \alpha' \) is a sparse vector. We assume \( \alpha' \) has \( n \) elements and its sparsity is \( k \). If the attacker wants to get \( \alpha' \) from \( \alpha^* \), then he should consider all the possibilities:

\[
C(n,k) \times n \times (n-1) \cdots (n-k+1) = C(n,k) \times \frac{n!}{(n-k)!}
\]  

where \( C(n,k) = n! / (n-k)!k! \). Since the attacker does not know \( k \), he has to consider all the possibilities:

\[
\sum_{k=1}^{n} C(n,k) \times \frac{n!}{(n-k)!} \quad (13)
\]

So, the probability that an attacker can get \( \alpha' \) from \( \alpha^* \) is:
According to (14), when $n$ is large, the probability that an attacker can get $\alpha'$ from $\alpha^*$ is extremely small, so it is difficult for an attacker to get information about the original data from the data reconstructed by cloud. Of course, no watermark information can be obtained. To sum up, our encryption scheme has certain security and can provide good privacy protection for data.

In the process of encryption, since we do not discuss the watermark parameters temporarily, let $d = 20, t = 50$, and the length of watermark be 640 bits in the whole image. Lena and Peppers in Figure 2(a)(e) has been chosen as the original image for performance evaluation. Figure 2(b)(f) is the reconstructed image by cloud without decryption. If the attacker or cloud has no encryption key, they cannot get the original image or some information about the original image. So our scheme is privacy-assured. Figure 2(c)(g) is the decrypted image with watermark by the receiver with correct encryption key. We can see that the recovered image is almost the same as the original image, but we can find some ripple in it compared with the original image. Figure 2(d)(h) is the decrypted image without watermark. There is no ripple in it and its quality is higher than Figure 2(c)(g).

### Data Hiding Performance

In our experiment, peak signal to noise ratio ($PSNR$) and bit-correct rate ($BCR$) are used as metrics to evaluate the quality of the reconstructed image and watermark. The definitions are as follows:

\[
PSNR = 10 \times \log_{10} \left( \frac{(2^n - 1)^2}{MSE} \right)
\]  

\[
BCR = 1 - \frac{error\ bits}{embedding\ amount}
\]  

**Figure 2a. Demonstration of privacy-assurance in our proposed system. (a) Original image**
Figure 2b. Demonstration of privacy-assurance in our proposed system. (b) Reconstructed image by cloud without decryption

Figure 2c. Demonstration of privacy-assurance in our proposed system. (c) Decrypted image with watermark

Figure 2d. Demonstration of privacy-assurance in our proposed system. (d) Decrypted image without watermark
Figure 2e. Demonstration of privacy-assurance in our proposed system. (e) Original image

Figure 2f. Demonstration of privacy-assurance in our proposed system. (f) Reconstructed image by cloud without decryption

Figure 2g. Demonstration of privacy-assurance in our proposed system. (g) Decrypted image with watermark
where MSE is mean square error. We first set parameter $d = 20$ and the embedding amount to be 10-bit per block to evaluate the effect of parameter on data hiding. The result is shown in Table 1, and the data in the table is the average of the ten images. PSNR1 is between the original images and the reconstructed images with watermark, PSNR2 is between the original images and the reconstructed images after extracting watermark.

Before the experiment, we randomly selected 20 blocks and got the $\alpha$ value of each block. We found that nearly 98% of the last 512 element absolute values in $\alpha$ are between 0-20, while 90% of the element absolute values are between 0-5. In order to extract the watermark more accurately, we start from $t = 30$.

We also plot the sparse data $\alpha$ and $\alpha'$ in Figure 3. Figure 3(a) is $\alpha$ obtained by the original CS process through (4). Figure 3(b) is $\alpha'$ obtained by our proposed scheme when $t=30$. 3(c) is $\alpha'$ obtained by our proposed scheme when $t=70$.

In Table 1, with the increase of $t$, PSNR1 is getting less, but PSNR2 remains about 35.6. After extracting the watermark, the reconstructed image can hold high quality, and the reconstructed image with watermark can also be accepted for some situations that do not request high image quality. From error bits and BCR, it can also be found that when $t=60$ or bigger, this setting is suitable for our scheme.

The function of $d$ is to hold the sparsity of watermark and control the embedding capacity. Based on the above discussion, we set $t=60$ to process this step. The experiment result is shown in Table 2. It can be concluded that the performance of our scheme is fine when the embedding amount is 640bits and $d \leq 20$, but when $d = 30$, the extraction has some errors. When the value of $d$ is large, the sparsity of the sparse signal is greatly affected, which reduces the reconstruction quality, resulting in extraction error. The same things happen when the embedding amount equals 1280 bits and $d = 20$.

Table 1. The Different PSNR/ Error Bits and BCR with Different Parameter $t$

| $t$  | PSNR1(dB) | PSNR2(dB) | error bits | BCR      |
|-----|-----------|-----------|------------|----------|
| t=30| 34.8521   | 35.6372   | 10         | 98.4375% |
| t=40| 34.4232   | 35.5652   | 2          | 99.6875% |
| t=50| 33.8720   | 35.6754   | 1          | 99.8438% |
| t=60| 32.9918   | 35.6017   | 0          | 100%     |
| t=70| 32.6896   | 35.5982   | 0          | 100%     |
Figure 3a. Demonstration of the back position of two sparse data $\alpha$ and $\alpha'$. (a): $\alpha$

Figure 3b. Demonstration of the back position of two sparse data $\alpha$ and $\alpha'$. (b): $\alpha'$ when $t=30$
Efficiency Evaluation

Let us focus on computation cost at data holder and user in the following cases: (1) the original CS image process without cloud-assistance and security consideration (Original_CS), (2) our proposed scheme only with encryption (Cloud_Non_Watermark), and (3) our proposed scheme with both watermark and encryption.

For ease of presentation, we implement different schemes on the image blocks with the same sizes, 24×24 or 32×32. All the results represent the mean of 50 experiments and each trial focuses on one randomly selected image block recovery. For Original_CS, $t_{DH}$ is the time for image sampling and compression, while $t_{user}$ is the time for CS reconstruction and image recovery. For Cloud_Non_Watermark, $t_{DH1}$ is the time for image sampling, compression and encryption, while $t_{user1}$ is the time for image decryption and image recovery. For our proposed scheme, $t_{DH2}$ is the time for image

| Embedding amount | d   | PSNR1(dB) | PSNR2(dB) | error bits | BCR      |
|------------------|-----|-----------|-----------|------------|----------|
| 640bit           | d=5 | 33.2491   | 35.6507   | 0          | 100%     |
|                  | d=10| 33.2819   | 35.7022   | 0          | 100%     |
|                  | d=20| 33.2516   | 35.6117   | 0          | 100%     |
|                  | d=30| 33.1025   | 35.5661   | 2          | 99.6875% |
| 1280bit          | d=5 | 32.6996   | 35.5309   | 0          | 100%     |
|                  | d=10| 32.0561   | 35.5906   | 0          | 100%     |
|                  | d=20| 31.9605   | 35.5102   | 2          | 99.8438% |
sampling, compression, encryption and watermark embedding, $t_{\text{user2}}$ is the time for image decryption, image recovery and watermark extraction.

Table 3 shows the mean running time. Similar to (Wang et al., 2014) and (X. Wu et al., 2014), data holder needs to do extra computation for security. Therefore, for Cloud_Non_Watermark, security computation cost on data holder increases by a factor of 1.8× ~2.0×; while for our scheme, running time only increases 2.3× ~2.5×. In (Wang et al., 2014) and (X. Wu et al., 2014), the computation costs on data holder increase by a factor of 5.6× ~9.4× and 1.2 ~1.7 ×, respectively. As can be seen from Table 3, our scheme can get low-complexity in data holder compared with (Wang et al., 2014) and just a little higher than (X. Wu et al., 2014). We also evaluate the total efficiency from Table 3. The last two columns, $t_1 = (t_{\text{DH}} + t_{\text{user}}) / (t_{\text{DH1}} + t_{\text{user1}})$ and $t_2 = (t_{\text{DH}} + t_{\text{user}}) / (t_{\text{DH2}} + t_{\text{user2}})$, show that Cloud_Non_Watermark can decrease 4.1× ~4.2× time cost and our proposed scheme can decrease 3.5× ~3.9×, so the efficiency impact of watermark is little. In (Wang et al., 2014), the authors have not contained the final image recovery cost in user. They decrease the total running time 3.4× ~4.0×. In (X. Wu et al., 2014), they decrease the total running time 4.1× ~4.2×. The experiment results show that our scheme can keep low-complexity and provide double protection compared with the existing cloud-assisted image service schemes.

Robustness

In the proposed scheme, a large amount of data is transmitted over the public channels, where noise may be generated, so the watermark scheme must have a certain degree of robustness to enable the data to be transmitted successfully. In the following experiment, it is assumed that the transmitted data is attacked by Gaussian white noise of different degrees. The experimental results are shown in Figure 4. Here, the test parameters are set to $t = 60, d = 20$ and the overall embedding amount is set to 640bit.

The four images in Figure 4 are decrypted images after watermark extraction. As we can see, with the increase of noise, the quality of the decrypted image decreases significantly. However, when the noise level is low, relatively clear images can still be obtained. At the same time, it can be seen that in the case of noise attack, the watermarking scheme in this paper can still maintain a high extraction accuracy. Only when the variance $\sigma^2$ of noise is greater than 7, the watermark is extracted with a deviation, among which, there is a 1-bit error in Figure 4(c) and a 2-bit error in Figure 4(d). However, the intensity of the Gaussian white noise used in the experiment has far exceeded the noise generated by the actual channel, so the watermarking scheme proposed in this paper has certain robustness.

CONCLUSION

In this paper, a cloud-assisted double protection scheme with encryption and data hiding based on CS is proposed. Taking better efficiency and security into consideration, this scheme exploits techniques

| Block size | Original CS | Our scheme only with encryption | Our integral scheme | Speedup |
|------------|-------------|--------------------------------|---------------------|---------|
|            | $t_{\text{DH}}$ | $t_{\text{user}}$ | $t_{\text{DH1}}$ | $t_{\text{user1}}$ | $t_{\text{DH2}}$ | $t_{\text{user2}}$ | $t_1$ | $t_2$ |
| 24 × 24    | 0.0087      | 0.2491            | 0.0162             | 0.0467             | 0.0201             | 0.0531             | 4.1 × | 3.5 × |
| 32 × 32    | 0.0293      | 1.0038            | 0.0586             | 0.1879             | 0.0732             | 0.1917             | 4.2 × | 3.9 × |
Figure 4a. Gauss white noise attack in different degrees

Figure 4b. Gauss white noise attack in different degrees

Figure 4c. Gauss white noise attack in different degrees

Figure 4d. Gauss white noise attack in different degrees
from different domains. The data holder performs data encryption and data hiding in samples which are sampled by devices with constrained resources. That is to say, this part is carried out in the compressive sensing domain, while the users can perform data decryption and watermark extraction in sparse domain directly. This can greatly reduce the amount of calculation of user. The watermark can be safely transferred among the data holder, cloud and user, and this scheme is privacy-assured where cloud knows neither the original data nor the watermark. Simulation results have been provided to demonstrate the privacy-protection, efficiency, accuracy of watermark of our scheme.

ACKNOWLEDGMENT

The work was funded by the National Natural Science Foundation of China (Grant Nos. 61572089, 61633005), the Chongqing Research Program of Basic Research and Frontier Technology (Grant No. cstc2017jcyjBX0008), the Project Supported by Graduate Student Research and Innovation Foundation of Chongqing (Grant No. CYB17026), the Chongqing Postgraduate Education Reform Project (Grant No. yjg183018) and the Chongqing University Postgraduate Education Reform Project (Grant No. cquyjg18219).
REFERENCES

Candes, E., & Romberg, J. (2006). Sparsity and incoherence in compressive sampling. Inverse Problems, 23(3), 969-985.

Candes, E., & Tao, T. (2006). Near-Optimal Signal Recovery From Random Projections: Universal Encoding Strategies? IEEE Transactions on Information Theory, 52(12), 5406–5425. doi:10.1109/TIT.2006.885507

Donoho, D. (2006). Compressed sensing. IEEE Transactions on Information Theory, 52(4), 1289–1306. doi:10.1109/TIT.2006.871582

Estrin, D., Govindan, R., Heidemann, J., & Kumar, S. (1999). Next century challenges: scalable coordination in sensor networks. Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking. doi:10.1145/313451.313556

Huang, K., Xu, M., Fu, S., & Wang, D. (2017). Practical privacy-preserving compressed sensing image recovery in the cloud. Science China. Information Sciences, 60(9), 098103. doi:10.1007/s11432-016-9055-2

Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. Proceedings of the sixteenth annual ACM symposium on Theory of computing. doi:10.1145/800057.808695

Orsdemir, A., Altun, H. O., Sharma, G., & Bocko, M. F. (2008). On the security and robustness of encryption via compressed sensing. Paper presented at the MILCOM 2008 - 2008 IEEE Military Communications Conference.

Pan, J.-S., Li, W., Yang, C.-S., & Yan, L.-J. (2015). Image steganography based on subsampling and compressive sensing. Multimedia Tools and Applications, 74(21), 9191–9205. doi:10.1007/s11042-014-2076-1

Peng, H., Tian, Y., & Kurths, J. (2017, March 22). Semitensor Product Compressive Sensing for Big Data Transmission in Wireless Sensor Networks. Mathematical Problems in Engineering, 2017, 1–8. doi:10.1155/2017/8158465

Rachlin, Y., & Baron, D. (2008). The Secrecy of Compressed Sensing Measurements. Paper presented at the Communication, Control, and Computing, 2008 Allerton Conference on. doi:10.1109/ALLERTON.2008.4797641

Valenzise, G., Tagliasacchi, M., Tubaro, S., Cancelli, G., & Barni, M. (2009). A compressive-sensing based watermarking scheme for sparse image tampering identification. Paper presented at the 2009 16th IEEE International Conference on Image Processing (ICIP).

Wang, C., Ren, K., & Wang, J. (2011). Secure and practical outsourcing of linear programming in cloud computing. INFOCOM, 2011 Proceedings IEEE. doi:10.1109/INFCom.2011.5935305

Wang, C., Zhang, B., Ren, K., & Roveda, J. M. (2014). A privacy-aware cloud-assisted healthcare monitoring system via compressive sensing. Paper presented at the IEEE INFOCOM 2014 - IEEE Conference on Computer Communications. doi:10.1109/INFOCom.2014.6848155

Wu, D., Yang, B., Wang, C., Wang, R., & Wang, R. (2016). Privacy-Preserving Multimedia Big Data Aggregation in Large-Scale Wireless Sensor Networks. ACM Transactions on Multimedia Computing Communications and Applications, 12(4s), 60. doi:10.1145/2978570

Wu, X., Tang, S., & Yang, P. (2014). Low-Complexity Cloud Image Privacy Protection via Matrix Perturbation. Academic Press.

Xiao, D., Cai, H., Wang, Y., & Bai, S. (2016). High-capacity separable data hiding in encrypted image based on compressive sensing. Multimedia Tools and Applications, 75(21), 13779–13789. doi:10.1007/s11042-015-2922-9

Xie, K., Ning, X., Wang, X., He, S., Ning, Z., Liu, X., & Qin, Z. (2016). An efficient privacy-preserving compressive data gathering scheme in WSNs. Information Sciences, 180(2), 702–715.
Di Xiao (PhD) received his B.S. degree from Sichuan University, Chengdu, China, and his M.S. and Ph.D. degrees from Chongqing University, Chongqing, China. From 2006 to 2008, he accomplished the postdoctoral research in Chongqing University. From December 2008 to December 2009, he visited the Department of Computer Science, New Jersey Institute of Technology, USA. He is currently a Full Professor with the College of Computer Science, Chongqing University, Chongqing, China. His research interests cover Signal Processing in Encrypted Domain, Compressive Sensing, Security and Privacy in Internet of things, etc. So far, he has published more than 90 academic journal papers and be successively elected as 2014-2018 Elsevier Most Cited Chinese Researchers.

Jia Liang received the Bachelor's degree from Qinghai University, Qinghai, China in 2016. Currently, she is pursuing her Ph. D. degree in Computer Science and Technology at the College of Computer Science, Chongqing University, China. Her research interests include Compressive Sensing, Machine Learning, Signal Processing in Encrypted Domain, etc.

Yanping Xiang received the Bachelor’s degree from the College of Computer Science, Chongqing University, China in 2014. Now she is pursuing her PhD. degree in computer science and technology at the College of Computer Science, Chongqing University, China. Her research interests include watermarking, signal processing over encrypted domain, etc.

Jiaqi Zhou received his M.S. degrees from Chongqing University, Chongqing, China in 2016. His research interests include Compressive Sensing, Signal Processing in Encrypted Domain, etc.