**SCIENCE CHINA**
Information Sciences

• **Supplementary File** •

# Practical privacy-preserving compressed sensing image recovery in the cloud

Kai HUANG[1,2], Ming XU[1], Shaojing FU[1,2*] & Dongsheng WANG[1]

[1]*College of Computer, National University of Defense Technology, Changsha* 410000*, China;*
[2]*State Key Laboratory of Cryptology, P.O.Box 5159, Beijing* 100000*, China*

## Appendix A Preliminaries

In the following, we present some basic terminologies and algorithms that are adopted in our scheme.

### Appendix A.1 Compressed Sensing

Compressed sensing (CS), also known as compressive sensing or compressed sampling, is a novel sensing/sampling paradigm that goes against the common wisdom in data acquisition. CS asserts that one can recover certain signals from far fewer samples or measurements than traditional methods use [1]. The underlying assumption that makes it possible is sparsity.

Given a vector $\boldsymbol{f} \in \mathbb{R}^n$, it can be expressed in terms of an orthonormal basis $\boldsymbol{\Psi}$ as $\boldsymbol{f} = \boldsymbol{\Psi x}$, where $\boldsymbol{x}$ is the vector of transformation coefficients and $\boldsymbol{x} = \boldsymbol{\Psi}^T \boldsymbol{f}$. Then $\boldsymbol{f}$ is said to be sparse if the coefficients are mostly zero or close to zero. And the sparsity of $\boldsymbol{x}$ in terms of $\boldsymbol{\Psi}$ is quantified by the number of significant (nonzero) coefficients, $k$. We measure the signal by taking inner products with a set of $m$ vectors $\boldsymbol{\Phi} : \{\boldsymbol{\phi}_i, i = 1, 2, \cdots, m\}$ that are incoherent with the sparsity basis $\boldsymbol{\Psi}$. Here we refer to $\boldsymbol{\Phi}$ as the sensing matrix, and by incoherent we mean that none of the vectors in $\boldsymbol{\Phi}$ have a sparse or compressible representation in terms of $\boldsymbol{\Psi}$. Then we can obtain the $m$ random measurements $\boldsymbol{y} = \boldsymbol{\Phi f} = \boldsymbol{\Phi \Psi x} = \boldsymbol{\Theta x}$, where $\boldsymbol{\Theta} = \boldsymbol{\Phi \Psi}$ is a $m \times n$ measurement matrix. Seminal results in compressed sensing show that it is possible to recover a $k$-sparse signal $\boldsymbol{x}$ from $m = O(klogn/k)$ linear measurements, as long as the measurement matrix $\boldsymbol{\Theta}$ is chosen to satisfy the restricted isometry property (RIP) [2].

There are two broad classes of techniques to recover the sparse signal from its unique measurements, Basis Pursuit (BP) [3] and Matching Pursuit (MP) [4]. For the BP method, on the one hand, it may take a long time to solve the linear program even for signals of moderate length. On the other hand, when off-the-shelf optimization software is not available, the implementation of optimization algorithms may demand serious effort. However, numerical experiments have demonstrated good performance using MP for reconstruction even though there are no theoretical guarantee. Therefore, we adopt the popular MP method in our paper. The matching pursuits iteratively identify the nonzero indices of $\boldsymbol{x}$. One of the fundamental matching pursuit techniques is Orthogonal Matching Pursuit (OMP).

### Appendix A.2 Orthogonal Matching Pursuit

In the problem of CS recovery using OMP, it is known a priori that the measured signal $\boldsymbol{x}$ is $k$-sparse, which means $\boldsymbol{x}$ has non-zero entries only at $k$ unknown indices. Let $\Omega$ be the support set that composed of the locations of all non-zero entries of $\boldsymbol{x}$, then $||\boldsymbol{x}||_0 = |\Omega| = k$. We refer to the columns $\boldsymbol{\theta}_j$ in $\boldsymbol{\Theta}$ corresponding to the indices $j \in \Omega$ as correct columns, and rest $\boldsymbol{\theta}_j : j \notin \Omega$ as wrong columns. It has been shown that OMP will exactly recover the support set of a sparse signal $\boldsymbol{x}$ from the measurement vector $\boldsymbol{y} = \boldsymbol{\Theta x}$, if certain requirements are satisfied with the coherence parameter or the restricted isometry property [5,6].

The key idea of OMP lies in the attempt to reconstruct the support set $\Omega$ of $\boldsymbol{x}$ iteratively by starting with $\Omega_0 = \emptyset$. At iteration $t$, the inner products between the columns of $\boldsymbol{\Theta}$ and the residual $\boldsymbol{r}_{t-1}$ are calculated, and the index of the largest absolute value of inner products is added to $\Omega$. That is to find the column that is most strongly correlated with the residual $\boldsymbol{r}_{t-1}$. Here, the residual $\boldsymbol{r}_{t-1}$ from the former iteration represents the component of the measurement vector $\boldsymbol{y}$ that cannot be spanned by the columns of $\boldsymbol{\Theta}$ indexed by $\Omega$. That means the residual $\boldsymbol{r}_{t-1}$ is always orthogonal to all the selected columns $\boldsymbol{\Theta}_{\Omega_{t-1}}$. Thus at iteration $t$, OMP will select an column that is linearly independent from the previously

* Corresponding author (email: shaojing1984@163.com)

**Algorithm A1** OMP algorithm

**Input:** $\boldsymbol{y} \in \mathbb{R}^m$ and $\boldsymbol{\Theta} \in \mathbb{R}^{m \times n}$;
**Output:** $\boldsymbol{x} \in \mathbb{R}^n$;

1: Initialize the residual $\boldsymbol{r}_0 = \boldsymbol{y}$ and the set of selected column $\Omega_0 = \emptyset$. Let the iteration counter $t = 1$.
2: Find the index $j_t$ that solves the easy optimization problem $j_t = arg\max_j |\langle \boldsymbol{r}_{t-1}, \boldsymbol{\theta}_j \rangle|$.
3: Augment the index set $\Omega_t = \Omega_{t-1} \cup \{j_t\}$.
4: Solve a least-squares problem $\min_{\boldsymbol{x}} \|\boldsymbol{y} - \boldsymbol{\Theta}_{\Omega_t}\boldsymbol{x}\|_2$ to obtain a new signal estimate $\boldsymbol{x}_t = (\boldsymbol{\Theta}_{\Omega_t}^H \boldsymbol{\Theta}_{\Omega_t})^{-1}\boldsymbol{\Theta}_{\Omega_t}^H \boldsymbol{y}$, where $\boldsymbol{\Theta}_{\Omega_t} = [\boldsymbol{\theta}_{j_1}, \cdots, \boldsymbol{\theta}_{j_t}]$, $j_1, \cdots, j_t \in \Omega_t$.
5: Update the residual $\boldsymbol{r}_t = \boldsymbol{y} - \boldsymbol{\Theta}_{\Omega_t}\boldsymbol{x}_t$.
6: Increment $t$, and return to Step 2 if $t < k$, where $k$ is the sparsity of the signal.
7: The estimate $\widehat{\boldsymbol{x}}$ for the ideal signal has nonzero indices at the components listed in $\Omega_k$. The value of the estimate $\widehat{\boldsymbol{x}}$ in components $j_t$ equals the $t$-th component of $\boldsymbol{x}_t$.

selected columns $\boldsymbol{\Theta}_{\Omega_{t-1}}$. In this way, the columns of $\boldsymbol{\Theta}$ that are "the most relative" to $\boldsymbol{y}$ are iteratively selected. The procedure is described in Algorithm A1.

## Appendix A.3 Garbled Circuits

Secure two-party computation enables two parties to compute a function collaboratively without either party learning anything other than the output of the function. In the 1980's, Yao presented the first general solution using garbled circuits for the problem of secure two-party computation in the presence of semi-honest adversaries.

Let $f$ be a polynomial-time function, and $x_1$ and $x_2$ be the respective inputs of the two parties $P_1$ and $P_2$. Yao's garbled circuits protocol first models the function $f$ as a Boolean circuit $C$ that will be computed gate by gate, from the input wires to the output wires. Then $P_1$ garbles the truth table for each gate in $C$ and generate a garbled version of the circuit, $C_g$. This is accomplished by replacing all Boolean values except the final outputs in $C$ with identically distributed pseudo-random values. In this way, all gates in the circuit become a function mapping two random input values to a random output value. And the mapping should have the property that given two input values of a gate, it is only possible to learn the output value that corresponds to the output of the gate (the other output value must be kept secret) [7]. To facilitate the secure computation, on the one hand, $P_1$ sends to $P_2$ the random value corresponding to his input Boolean value; on the other hand, $P_2$ engages in a 1-out-of-2 oblivious transfer (OT) protocol with $P_1$ to obtain the random value corresponding to her input Boolean value. Thus, $P_1$ does not know $P_2$'s inputs, nor does $P_2$ know $P_1$'s input. Once $P_2$ has the garbled circuit and both garbled inputs, she can straight forwardly compute the entire circuit $C_g$. And the outputs will be finally shared by $P_1$ and $P_2$.

In this paper, $f$ refers to a simple comparison function. For two inputs $x_1$ and $x_2$, $f$ will output 1 if $x_1 \geqslant x_2$; otherwise output 0. Although Yao's original approach was too computationally expensive for practical use, a great deal of work has gone into optimizing the protocol in the past decades [8–10]. Especially, the recently proposed Half Gates method [10] can significantly reduce the total time and energy of the garbled circuit technique. Therefore, we employ it in our scheme to implement the secure outsourcing image recovery.

## Appendix B The Collaborative OMP Protocol

See Algorithm B1 for the details of our collaborative OMP protocol, where we use $f_1$ and $f_2$ to distinguish between the two rounds of comparisons.

## Appendix C Theoretical Analysis

### Appendix C.1 Correctness Analysis

In this subsection, we will provide a theoretical examination of the correctness of our scheme.

In our scheme, to encrypt the acquired compressed sensing signal, $\boldsymbol{y}$ is divided into two parts by $\boldsymbol{y} = \boldsymbol{y}' - \boldsymbol{y}''$. Intuitively, for a underdetermined system $\boldsymbol{y} = \boldsymbol{\Theta}\boldsymbol{x}$, even if $\boldsymbol{y}' = \boldsymbol{\Theta}\boldsymbol{x}'$ and $\boldsymbol{y}'' = \boldsymbol{\Theta}\boldsymbol{x}''$, we don't necessarily have $\boldsymbol{x} = \boldsymbol{x}' - \boldsymbol{x}''$. However, through our collaborative design, we can ensure that the final result satisfies the above equation.

In step 1, we initialize the residual $\boldsymbol{r}_0{}' = \boldsymbol{y}'$ and $\boldsymbol{r}_0{}'' = \boldsymbol{y}''$ in the two cloud servers $\mathcal{S}_1$ and $\mathcal{S}_2$, respectively. Then we have $\boldsymbol{r}_0 = \boldsymbol{r}_0{}' - \boldsymbol{r}_0{}'' = \boldsymbol{y}' - \boldsymbol{y}''$.

In step 2, since the most correlated column is determined collaboratively by $\mathcal{S}_1$ and $\mathcal{S}_2$, the result are same in $\mathcal{S}_1$ and $\mathcal{S}_2$. Therefore, $\boldsymbol{\Theta}_{\Omega_t}$ are consistent in $\mathcal{S}_1$ and $\mathcal{S}_2$.

In step 3, since $\boldsymbol{\Theta}_{\Omega_t}$ are consistent in $\mathcal{S}_1$ and $\mathcal{S}_2$, the solution of the minimization problem $\min_x \|\boldsymbol{y} - \boldsymbol{\Theta}_{\Omega_t}\boldsymbol{x}\|$ can be acquired by computing

$$\begin{aligned}\boldsymbol{x}_t &= (\boldsymbol{\Theta}_{\Omega_t}^H \boldsymbol{\Theta}_{\Omega_t})^{-1}\boldsymbol{\Theta}_{\Omega_t}^H \boldsymbol{y} \\ &= (\boldsymbol{\Theta}_{\Omega_t}^H \boldsymbol{\Theta}_{\Omega_t})^{-1}\boldsymbol{\Theta}_{\Omega_t}^H (\boldsymbol{y}' - \boldsymbol{y}'') \\ &= \boldsymbol{x}_t{}' - \boldsymbol{x}_t{}''\end{aligned}$$

**Algorithm B1** Collaborative OMP

**Input:** $\boldsymbol{y}' \in \mathbb{R}^m$, $\boldsymbol{y}'' \in \mathbb{R}^m$, and $\boldsymbol{\Theta} \in \mathbb{R}^{m \times n}$;
**Output:** $\boldsymbol{x}' \in \mathbb{R}^n$, $\boldsymbol{x}'' \in \mathbb{R}^n$;

$\mathcal{S}_1$ and $\mathcal{S}_2$ initialize the residual $\boldsymbol{r}_0{}' = \boldsymbol{y}'$ and $\boldsymbol{r}_0{}'' = \boldsymbol{y}''$, respectively; and they both initialize the set of selected column $\Omega_0 = \emptyset$. Let the iteration counter $t = 1$. Repeat the following steps until the stop criterion is satisfied.
**for** $i = 1$ to $n$ **do**
  $\mathcal{S}_1$ computes $p_i{}' = \langle \boldsymbol{r}_{t-1}{}', \boldsymbol{\theta}_i \rangle$ and sends them to $\mathcal{S}_2$ in encrypted form;
  $\mathcal{S}_2$ computes $p_i{}'' = \langle \boldsymbol{r}_{t-1}{}'', \boldsymbol{\theta}_i \rangle$ and obtains the encrypted form from $\mathcal{S}_1$ through oblivious transfer;
  $\mathcal{S}_2$ computes $f_1(p_i{}', p_i{}'')$ using the garbled circuit;
**end for**
**for** $i = 1$ to $n - 1$ **do**
  **for** $j = i + 1$ to $n$ **do**
    $\mathcal{S}_1$ computes $p_i{}' + p_j{}'$ and $p_i{}' - p_j{}'$ and sends them to $\mathcal{S}_2$ in encrypted form;
    $\mathcal{S}_2$ computes $p_i{}'' + p_j{}''$ and $p_i{}'' - p_j{}''$ and obtains the encrypted form from $\mathcal{S}_1$ through oblivious transfer;
  **end for**
**end for**
$j_t = 1$;
**for** $i = 2$ to $n$ **do**
  **if** $f_1(p_i{}', p_i{}'') = 1$ and $f_1(p_{j_t}{}', p_{j_t}{}'') = 1$ **then**
    $\mathcal{S}_2$ computes $f_2(p_i{}' - p_{j_t}{}', p_i{}'' - p_{j_t}{}'')$ using the garbled circuit;
  **else if** $f_1(p_i{}', p_i{}'') = 1$ and $f_1(p_{j_t}{}', p_{j_t}{}'') = 0$ **then**
    $\mathcal{S}_2$ computes $f_2(p_i{}' + p_{j_t}{}', p_i{}'' + p_{j_t}{}'')$ using the garbled circuit;
  **else if** $f_1(p_i{}', p_i{}'') = 0$ and $f_1(p_{j_t}{}', p_{j_t}{}'') = 1$ **then**
    $\mathcal{S}_2$ computes $f_2(p_i{}'' + p_{j_t}{}'', p_i{}' + p_{j_t}{}')$ using the garbled circuit;
  **else**
    $\mathcal{S}_2$ computes $f_2(p_i{}'' - p_{j_t}{}'', p_i{}' - p_{j_t}{}')$ using the garbled circuit;
  **end if**
  **if** $f_2 = 1$ **then**
    $j_t = i$;
  **end if**
**end for**
$\Omega_t = \Omega_{t-1} \cup \{j_t\}$;
$\boldsymbol{\Theta}_{\Omega_t} = [\boldsymbol{\theta}_{j_1}, \cdots, \boldsymbol{\theta}_{j_t}]$, $j_1, \cdots, j_t \in \Omega_t$;
$\mathcal{S}_1$ computes $\boldsymbol{x}_t{}' = (\boldsymbol{\Theta}_{\Omega_t}^H \boldsymbol{\Theta}_{\Omega_t})^{-1} \boldsymbol{\Theta}_{\Omega_t}^H \boldsymbol{y}'$. Update $\boldsymbol{r}_t{}' = \boldsymbol{y}' - \boldsymbol{\Theta}_{\Omega_t} \boldsymbol{x}_t{}'$;
$\mathcal{S}_2$ computes $\boldsymbol{x}_t{}'' = (\boldsymbol{\Theta}_{\Omega_t}^H \boldsymbol{\Theta}_{\Omega_t})^{-1} \boldsymbol{\Theta}_{\Omega_t}^H \boldsymbol{y}''$. Update $\boldsymbol{r}_t{}'' = \boldsymbol{y}'' - \boldsymbol{\Theta}_{\Omega_t} \boldsymbol{x}_t{}''$;
Set $t \leftarrow t + 1$.

In step 4, both $\mathcal{S}_1$ and $\mathcal{S}_2$ will update the residual. Therefore we have

$$\begin{aligned} \boldsymbol{r}_t &= \boldsymbol{y} - \boldsymbol{\Theta}_{\Omega_t} \boldsymbol{x}_t \\ &= (\boldsymbol{y}' - \boldsymbol{y}'') - \boldsymbol{\Theta}_{\Omega_t} (\boldsymbol{x}_t{}' - \boldsymbol{x}_t{}'') \\ &= \boldsymbol{r}_t{}' - \boldsymbol{r}_t{}'' \end{aligned}$$

This will also ensure the subsequent iteration process be correct.

Up to now, we can see that each signal $\boldsymbol{x}$ the data user derives by computing $\boldsymbol{x} = \boldsymbol{x}' - \boldsymbol{x}''$ is consistent with the original OMP algorithm. Therefore, the data user is able to reconstruct the original image signal correctly in this way.

In addition, we design an efficient result verification method. The image data user only needs to perform a simple matrix-vector multiplication to verify the correctness of the returned result from the two cloud servers.

## Appendix C.2 Efficiency Analysis

The compressed sensing framework entails a computationally expensive recovery process, therefore, in our scheme, we shift the image reconstruction task to the cloud side to make the compressed sensing technique much more practical. Compared with the transformation based design in [11], which needs to perform the time-consuming matrix-matrix operations, in our scheme, through the splitting based image encryption design, the sensor side and the user side only need to do simple addition and subtraction operations. Moreover, there will be no difference even for different sizes of images. Therefore, our design can reduce the computation burden of the sensor side and the end user side tremendously.

As for the storage overhead at the sensor side, we can save much more space than the scheme in [11]. They need to generate the random matrix to transform the original linear program each time, however, in our scheme, we consider the measurement matrix to be public and don't need extra space to implement the encryption of the signal.

At the cloud side, we must point out that the two cloud servers have to interact with each other to exchange the intermediate result and determine the most correlated column in the measurement matrix collaboratively. However, as noted above, we can reduce the number of rounds of interactions by computing the results in each iteration all at once. Then $\mathcal{S}_1$ can encrypt and transmit all the intermediate results to $\mathcal{S}_2$ based on the garbled circuits and oblivious transfer

(a) (b) (c) (d) (e)

**Figure D1** Images: (a) Moon surface; (b) Airport; (c) Cameraman; (d) Lena; (e) MRI Brain.

technique. Consequently, the number of rounds of interaction required in each iteration are a constant, i.e. 2. And the most costly part lies in the comparison process in $\mathcal{S}_2$. In each iteration, $\mathcal{S}_2$ will need to perform $(2n-1)$ comparisons. Note that this can be performed simultaneously in $\mathcal{S}_2$, therefore it is affordable for a cloud server that has large amounts of computation resources.

## Appendix C.3 Security Analysis

To prevent the cloud servers from learning the contents of the image signal, in the signal encryption step, the signal $\boldsymbol{y}$ is encrypted by splitting into two parts $\boldsymbol{y}'$ and $\boldsymbol{y}''$ which are sent to two independent cloud servers. However, only when y is given is it possible to reconstruct $\boldsymbol{x}$ from the underdetermined linear system of equations. In our scheme, no one except the sensor knows the signal $\boldsymbol{y}$. Although the measurement matrix is designed to be public in our scheme, given $\boldsymbol{y}'$ or $\boldsymbol{y}''$, no one can solve $\boldsymbol{x}$ at all. Therefore, revealing the measurement matrix does not affect the privacy preservation. And if the servers use the brute-force approach to recover the signal, for an image whose size is $256 \times 256$ and each pixel has 8 bits, it will need $256^{256\times256}$ operations and is computationally-infeasible in practice.

As for our collaborative OMP protocol, the secure comparison is implemented by using garbled circuit that is provably secure under our honest-but-curious model, so neither server can learn anything about each other's input. Both servers do know the result of each comparison, however, they are unable to compute the specific value of the signal at all. Therefore, we claim that the confidentiality of the sensed signal is well protected. Correspondingly, the cloud servers cannot recover the original image content either.

## Appendix D Empirical Evaluation

In this section, we evaluate the performance of our scheme on five grayscale images (Moon surface, Airport, Cameraman, Lena, and MRI Brain) of size $256\times256$ that are found frequently in the literature, as shown in Figure D1 . We implement the sensor/user side process in MATLAB and the cloud side protocol in the C language. And we implement our collaborative OMP protocol with the Obliv-C system [12] using the latest optimizations [10]. All experiments are done on the same workstation with an Intel Core i5 CPU running at 2.90 GHz and 6GB RAM. We only focus on the computational evaluations and ignore the communication latency between the image sensor, the end user, and the cloud servers.

In our work, instead of bothering with the imaging system, we simulate the compressed sensing process at the sensor side. The image can be stacked into one or several long vectors according to the lexicographical order, but here we treat the $256 \times 256$ image as 256 vectors. We generate the sensing matrix $\boldsymbol{\Phi}$ by sampling i.i.d. entries from the Gaussian distribution $N(0,1)$. As for the orthogonal basis $\boldsymbol{\Psi}$, it is well known that images may be represented sparsely in a discrete cosine basis as well as a wavelet basis [13]. Compared with the discrete cosine transform, the wavelet transform is a much sparser representation for photograph-like images. Moreover, it can be applied and inverted in $O(n)$ computations, by exploiting the multi-scale structure of the wavelet basis. This is even faster than the $O(nlogn)$ cost of the fast Fourier transform [14]. Therefore, we represent the images in a wavelet basis in our experiments. Then we derive $\boldsymbol{\Theta} = \boldsymbol{\Phi\Psi}$ that is used to reconstruct the sparse signal at the cloud side.

## Appendix D.1 Effectiveness Evaluation

We first assess the effectiveness of our scheme, including the correctness and the privacy assurance of our scheme.

Since there are many works to determine how many measurements $m$ are necessary to recover an $s$-sparse signal in $R^n$ with high probability. We also perform several trials with several measurement ratios, $m/n$. Since some studies have shown that the peak signal-to-noise ratio (PSNR) has the best performance in measuring the quality of noisy images, we measure the recovery performance in terms of PSNR between the recovered and original images. And the excellent range of values for the PSNR in lossy image compression is from 30dB to 50dB, provided the bit depth is 8 bits. As shown in Table D1, the reconstructed image quality increases along with the number of measurements and iterations. When $m = 180$ and $t = 56$, the PSNR mean value was 31.16dB, which becomes acceptable in evaluating the performance of the compressed sensing image recovery. Certainly, we must point out that the recovery effect would be better as $m$ and $t$ increases. However, we still adopt $m = 180$ at the sensor side and we set $t = 56$ at the cloud side in the following experiments. Figure D2 illustrates the example visual results when $m = 180$ and $t = 56$. Figure D2(a) gives the original "Lena" image, while Figure D2(f) depicts the recovery result based on the collaborative OMP protocol at the user side. We can see that fairly good performance can be obtained with such a configuration.

**Table D1** Effectiveness evaluation of our scheme in terms of PSNR (dB) at varying number of measurements and iterations ($n = 256$)

| Image | $m = 128$ | | | | $m = 154$ | | | | $m = 180$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | t=32 | t=40 | t=48 | t=56 | t=32 | t=40 | t=48 | t=56 | t=32 | t=40 | t=48 | t=56 |
| Moon surface | 27.84 | 28.02 | 27.87 | 27.50 | 28.83 | 28.82 | 28.91 | 29.09 | 29.30 | 29.63 | 29.89 | 29.98 |
| Airport | 32.56 | 34.08 | 34.04 | 33.71 | 33.49 | 35.73 | 36.36 | 37.30 | 34.39 | 36.20 | 37.60 | 38.90 |
| Cameraman | 23.02 | 23.26 | 23.18 | 23.27 | 24.27 | 24.88 | 25.01 | 25.34 | 24.87 | 25.77 | 26.39 | 26.96 |
| Lena | 25.45 | 25.62 | 25.59 | 25.42 | 26.81 | 27.47 | 27.58 | 27.73 | 27.23 | 28.09 | 28.60 | 29.06 |
| MRI Brain | 26.64 | 26.56 | 26.85 | 26.50 | 27.67 | 28.47 | 28.70 | 29.09 | 28.42 | 29.50 | 30.00 | 30.90 |

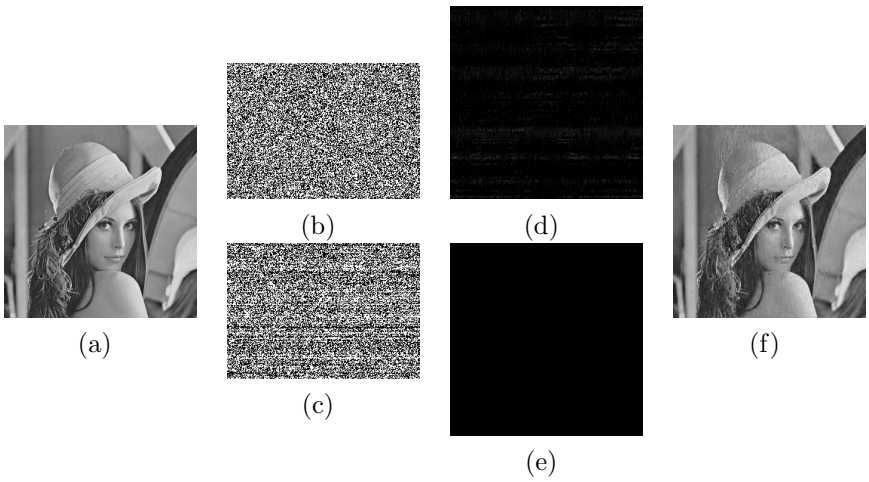


**Figure D2** Example image recovery of our proposed scheme ($m = 180, t = 56$): (a) Original Lena ($256 \times 256$); (b) $\boldsymbol{y}'$ sent to $\mathcal{S}_1$ ($180 \times 256$); (c) $\boldsymbol{y}''$ sent to $\mathcal{S}_2$ ($180 \times 256$); (d) $\boldsymbol{f}'$ reconstructed by $\mathcal{S}_1$ ($256 \times 256$); (e) $\boldsymbol{f}''$ reconstructed by $\mathcal{S}_2$ ($256 \times 256$); (f) Recovered Lena (PSNR=29.15, $256 \times 256$).

We also preserve the privacy of the images. To wit, Figure D2(b) presents the part $\boldsymbol{y}'$ for the "Lena" image that is outsourced to the cloud server $\mathcal{S}_1$ and Figure D2(d) illustrates the reconstructed $\boldsymbol{x}'$ by $\mathcal{S}_1$. Similarly, Figure D2(c) and Figure D2(e) show the part held by the cloud server $\mathcal{S}_2$. On the one hand, It is easy to see that $\mathcal{S}_1$ cannot obtain any useful information from the owned contents, and $\mathcal{S}_2$ cannot obtain the original signal either. On the other hand, since the two cloud servers are independent, they cannot get the corresponding part from the other one.

Therefore, by utilizing the two independent cloud servers, we can recover the images correctly while providing good enough privacy-assurance on the image content protection.

## Appendix D.2 Efficiency Evaluation

We next measure the efficiency of our scheme. We mainly focus on the computational cost of the work done by the sensor and the end user. The corresponding time costs are shown in Table D2 and all figures are averaged over 10 independent trials.

The second column displays the time cost of the sensor side, which includes the time to generate the random encryption vectors in addition to the time required by signal encryption before outsourcing. The third column displays the time cost of the end user side, which is mainly the decryption step. And we also include the time cost of the original recovery without outsourcing in the fourth column. Obviously, our scheme has extremely low time costs at the sensor side and the end user side which are nearly constant values. This implies that the most computationally intensive task is outsourced to the cloud servers in our scheme.

Then, we present the comparison of the asymmetric speedup (i.e., the total time cost of image recovery without outsourcing divided by the total time cost of image signal encryption and decryption at the sensor side and the end user side) between our scheme and Wang et al.'s scheme [11]. Recall that Wang et al.'s scheme focuses on one randomly selected image block recovery for each trial. Although we test the recovery of the whole image instead of the block of the image, we can still see that our scheme can achieve much more noticeable computation cost savings.

Besides, we compare the time cost with different image sizes. Compared with the original recovery process whose execution time grows linearly with the size of the images, there is little difference with regard to the time cost at the sensor side and the end user side in our scheme. These are due to the fact that we only need to take simple addition and subtraction operations at the sensor side and the end user side in our scheme.

**Table D2** Efficiency evaluation results of our scheme

| | Image size | Secure Outsourcing | | Without Outsourcing | Asymmetric Speedup |
|---|---|---|---|---|---|
| | | $t_{sensor}(ms)$ | $t_{user}(ms)$ | $t_{original}(s)$ | $\frac{t_{original}}{t_{sensor}+t_{user}}$ |
| Our scheme | $256 \times 256$ | 1.9 | 0.22 | 14.389 | 6755× |
| | $512 \times 512$ | 2.7 | 0.57 | 55.214 | 16885× |
| | $1024 \times 1024$ | 11.7 | 2.02 | 237.665 | 17322× |
| Wang et al.'s scheme[11] | $32 \times 32$ | 440 | 10 | 1.76 | 3.9× |
| | $48 \times 48$ | 4360 | 24 | 14.79 | 3.4× |

## References


1 Candès EJ, Wakin MB. An introduction to compressive sampling. IEEE Signal Processing Magazine, 2008, 25: 21-30
2 Candès EJ. The restricted isometry property and its implications for compressed sensing. Comptes Rendus Mathematique, 2008, 346: 589-592
3 Candès EJ, Tao T. Decoding by linear programming. IEEE Transactions on Information Theory, 2005, 51: 4203-4215
4 Tropp JA, Gilbert AC. Signal recovery from random measurements via orthogonal matching pursuit. IEEE Transactions on Information Theory, 2007, 53: 4655-4666
5 Ding J, Chen L, Gu Y. Perturbation analysis of orthogonal matching pursuit. IEEE Transactions on Signal Processing, 2013, 61: 398-410
6 Tropp JA, Wright SJ. Computational methods for sparse solution of linear inverse problems. Proceedings of the IEEE, 2010, 98: 948-958
7 Lindell Y, Pinkas B. A proof of security of Yao's protocol for two-party computation. Journal of Cryptology, 2009, 22: 161-188
8 Huang Y, Evans D, Katz J, et al. Faster secure two-party computation using garbled circuits. Usenix Conference on Security. 2011. 35-35
9 Bellare M, Hoang VT, Rogaway P. Foundations of garbled circuits. ACM Conference on Computer and Communications Security (CCS). 2013. 784-796
10 Zahur S, Rosulek M, Evans D. Two halves make a whole. Advances in Cryptology - EUROCRYPT 2015. 220-250
11 Wang C, Zhang B, Ren K, et al. Privacy-assured outsourcing of image reconstruction service in cloud. IEEE Transactions on Emerging Topics in Computing, 2013, 1: 166-177
12 Zahur S, Evans D. Obliv-C: A language for extensible data-oblivious computation. Cryptology ePrint Archive, Report 2015/1153, 2015
13 Baraniuk RG, Cevher V, Duarte MF, et al. Model-based compressive sensing. IEEE Transactions on Information Theory, 2010, 56: 1982-2001
14 Candès E, Romberg J. Sparsity and incoherence in compressive sampling. Inverse Problems, 2006, 23: 969-985