Hindawi
Security and Communication Networks
Volume 2017, Article ID 4376282, 14 pages
https://doi.org/10.1155/2017/4376282

Research Article

A Cloud-User Protocol Based on Ciphertext Watermarking Technology

Keyang Liu, Weiming Zhang, and Xiaojuan Dong

School of Information Science and Technology, University of Science and Technology of China, Anhui, China

Correspondence should be addressed to Weiming Zhang; zhangwm@ustc.edu.cn

Received 14 August 2017; Accepted 9 November 2017; Published 11 December 2017

Academic Editor: Kim-Kwang Raymond Choo

With the growth of cloud computing technology, more and more Cloud Service Providers (CSPs) begin to provide cloud computing service to users and ask for users' permission of using their data to improve the quality of service (QoS). Since these data are stored in the form of plain text, they bring about users' worry for the risk of privacy leakage. However, the existing watermark embedding and encryption technology is not suitable for protecting the Right to Be Forgotten. Hence, we propose a new Cloud-User protocol as a solution for plain text outsourcing problem. We only allow users and CSPs to embed the ciphertext watermark, which is generated and embedded by Trusted Third Party (TTP), into the ciphertext data for transferring. Then, the receiver decrypts it and obtains the watermarked data in plain text. In the arbitration stage, feature extraction and the identity of user will be used to identify the data. The fixed Hamming distance code can help raise the system's capability for watermarks as much as possible. Extracted watermark can locate the unauthorized distributor and protect the right of honest CSP. The results of experiments demonstrate the security and validity of our protocol.

1. Introduction

1.1. Problem Background. Right to Be Forgotten (RTBF) is a kind of people's right that was proposed for protecting people's privacy and has been mentioned as early as 1995 in Data Protection Directive of EU [1]. The 17th article of General Data Protection Regulation (GDPR) [2], which was passed by EU in 2012 to strengthen data protection for individuals in EU, defined RTBF as the right that people deserve to obtain or erase the data expired or related to their privacy from the data controller. In 2013, Senate Bill 568 of California [3] was signed to protect the RTBF of children. In 2014, the European Court of Justice compelled Google to delete the links about a Spanish man's bankruptcy from its searching results, which confirmed that the RTBF is a basic right for people living in EU. Since then, Google, Facebook, and YouTube have erased tens of thousands of links based on the request of EU citizens [4]. However, the erasure of data cannot be technically confirmed by users if they do not believe their Cloud Service Providers (CSPs). Moreover, cloud computing becomes more and more powerful and economic. Companies like Amazon, Alibaba, and Microsoft have provided cloud computing service to help people manipulating their data more cheaply and easily. If users want to lodge their data in cloud servers to lower the expenses, they need to think carefully about the risk of data leakage. As a result, confirmed deletion and several related ideas can be introduced to deal with this problem, which is also the target of our protocol.

1.2. Related Works. There are two kinds of methods used in confirmed deletion. First comes the encryption. User (U) encrypts his data and transfers it to CSP for storing [5–7]. Once U wants to delete his data, he just needs to abandon the encryption key and inform CSP that related data are useless. The management of key can be authorized to several Third Parties and use secret sharing technology to prevent conspiracy [8]. Encryption can protect the privacy of data and RTBF in ideal circumstance though it destroys the value of data. When U uses encryption technology, he can only use the storage space of CSP while wasting their ability of computation.

To solve this dilemma, homomorphic encryption (HE) [9] was introduced into this field [10]. Once the data is encrypted by HE algorithm, CSP can calculate data as user

ordered while knowing nothing about it. However, HE has some other flaws. For example, it requires user to have the knowledge about what operations they want to do on data before knowing their results. What is more, the full-HE, which can do both addition and multiplication on ciphertext, is unbearably slow and costs a lot. The semi-HE, which is faster, faces the problem of restricted operations. In a word, it is not convenient and economic for using encryption to protect RTBF so far. In our solution, data will be stored as plain text in cloud servers so that U can use the ability of computation completely to manage U's data.

Other than confirming deletion, not deleted is more easy to be confirmed, which suggests the second way, tracing the unauthorized distribution of data. To the best of our knowledge, watermarking is used in copy deterrence and tracing down the distribution of illegal copies [11–13]. This fact indicates watermark can be used to protect RTBF by proving the crimes of CSP. As the successful cases have shown, Google and Facebook were forced to delete [4] those links infringing people's privacy once U reports them and proves the infringement. But this method faces a new problem that user can use his data to fraud CSP if he can get benefit from lawsuit like defaming the specific CSP or diddling indemnity. If CSP requires embedding another watermark so that he can identify whether the copy is stored in his server, CSP can leak the copy with both watermarks to avoid being charged. Once the embedding process is outsourced to a Third Party, it will raise the risk of information leak from TP. In a word, watermark technology cannot be used to protect RTBF directly.

In this paper, we design a new Cloud-User protocol as the solution based on the work of buyer-seller protocol [14, 15]. We generate and embed the watermark in ciphertext to make sure the watermark can be erased during downloading. By using only one watermark, we increase the SNR of data. Moreover, we introduce the idea of feature-extraction function (FEF), a fixed Hamming distance code into protocol to reduce the cost of searching and increase the capacity of system while maintaining the robustness of watermark system.

The rest of this paper is organized as follows. In Section 2, we will give a brief introduction to the problem models, design goals, and the threat models. Preliminaries will be introduced in Section 3. The proposed solution is described in Section 4 and the security of the scheme is analyzed in Section 5. Section 6 will explain the design of experiment as well as the subalgorithm we used for building demo. The results of each experiment will be analyzed in Section 7. The last section contains the conclusions and future work.

2. Problem Formulation

2.1. Problem Model. The problem model in this paper involves three parties: User (U), Cloud Service Provider (CSP), and the Trusted Third Party (TTP).

User. U possesses large quantities of data and wishes to store it in the CSP's server. In addition, those data are valuable and need enough computing power to dig out their value. As a result, U wants to store his data on CSP's server and requires CSP to do some complex operations for him. In our scheme, only U and CSP can touch and manipulate these data. According to RTBF, U has the right to retrieve his data and require CSP to delete it at any time. Once U finds his data that should be deleted, U can suspect that a CSP has distributed his data illegally for interests and require TTP to verify where it comes from. Once confirmed, U can sue CSP for being guilty and ask for a compensation.

Cloud Service Provider. CSP controls piles of servers which have large storage space and powerful computing ability. U can store, manipulate, and delete his data on CSP's servers only if he pays for it according to contract. Although CSP controls all the data in his servers, he does not have the ownership of data and should take responsibility for their security. CSP can never distribute data whether U cares or not and needs to backup it in case of servers' crash.

Trusted Third Party. TTP is an arbitration agency who is responsible for generating a valid watermark for every single trade between U and CSP. TTP should be trusty so that his verification can be used as evidence. Besides, TTP should know nothing about U's data unless U requires TTP to verify whether a specific copy has been marked to be deleted.

Our solution is designed to make sure any one of the three can only know what they allowed to know and do what they required to do. Whoever disobeys the contract will suffer loss.

2.2. Threat Model. In the proposed solution, we assume TTP is selected by U and CSP, so we do not consider TTP will conspire with any one and no one can get payment from him. So there is no conspiracy among our solutions. We should consider the threat that CSP or U can get benefit while offending the other one.

CSP's Attack. CSP controls all data stored in his servers; he should obey U's order to manipulate U's data according to contract. But it may copy U's data as a backup even after U requires CSP to delete it. Since CSP has a full access to these data in plain text, we can do nothing about his analysis on data and that should be considered in contract. On one hand, CSP may not delete the data as required, and those data are leaked for CSP's careless management. On the other hand, CSP may deliberately sell these data after U's delete requirement and even try to adjust it so that U and TTP cannot trace it.

U's Attack. U possesses the ownership of data. The benefit U that can be gained from CSP is the compensation. On one hand, if CSP is innocent, U can only use the retrieved data and original data to create a copy. On the other hand, once CSP has leaked a part of U's data files, U may use them to guess other data files' watermark and forge CSP's loss.

2.3. Design Goals. This paper aims to design a solution among CSP, U, and TTP that allows U to store his data in CSP's servers as plain text while providing the remote control according to contract. In particular, we formally detail the goals as follows.

Data Privacy. As claimed in problem model, TTP is responsible for generating watermarks for giving data while TTP should have no access to data's content. Hence, we carefully design our solution so that TTP can embed the identity watermark directly in the encrypted data.

Nonrepudiation. Any copy of unauthorized data must be identifiable to find the illegal distributor.

Fairness. The proposed solution is secure and fair to all parties. Nobody can frame an honest party.

The Right to Be Forgotten. Acceptable deletion requires no information about the data remaining in servers of CSP. Once CSP does not follow requirements and the bad behavior can be proved by U (i.e., the unauthorized copy is detected). U can require TTP to verify the watermark of leaked copy and provide it as evidence which cannot be denied based on contract.

3. Preliminaries

In our solution, there are four kinds of technology we will use. Each technical method can be adjusted to fit all kinds of data (D). To simplify the declaration, we use image data as an example to introduce our solution and complete our experiment. Here is a simple introduction to these technologies and the restriction our solution required.

3.1. Feature-Extraction Function (FEF). FEF is used to identify the content of data while getting no detail about it. FEF is an important part of our solution which is used to define the validity of data for U. FEF's input is data file and the output is a feature (Fea). Once a data file A and its adjusted copy B satisfied FEF(A) = FEF(B), we call B a derivative copy (DC) of A. The set of DC is derivative set (DS).

In our solution, FEF must fit the following requirement.

(1) One-Way Function. For B = FEF(A), no one can create a DC of A if he only has the knowledge about B. This is because Fea of the stored data is shared among all three parties in our solution. This property can make sure only U and CSP can distribute DC of data.

(2) Content-Based. For no digital watermark algorithm can promise that it can resist all attacks, we use FEF as a restriction of watermark extraction algorithm so that our solution can get balance between validity and security. In our solution, U should carefully select FEF to make sure all the valuable copies of original data belong to a DS.

(3) Equiprobability. The set of possible value for Fea must be large enough, and the possibility of each value is equal. This property protects the efficiency of searching process.

3.2. Digital Watermark. Digital watermark (W) is a signal embedded into data to identify some attributions of the data (i.e., ownership). According to the domain embedded, digital watermark embedding algorithms are divided into time-spatial embedding, which is fast and relatively easy to operate but is easy to be erased by geometrical attack, and transform domain embedding [16–19], which is good at resisting geometrical attack but is fragile facing filtering. Moreover, according to the preknowledge related to data before embedding, we classify the embedding method into preknowledge dependent embedding and preknowledge independent embedding. In most cases, dependent embedding is more robust than independent one. In our solution, we recommend to use the preknowledge dependent transform embedding method to enhance the security of our solution. Furthermore, our solution requires the following properties that digital watermark embedding algorithm should have.

(1) Markov Property. For a given $W = w_i \mid 1 < i < L$ of length L, the embedding and extracting process of w_i has no effect on the process of w_{i+1}.

(2) Predictability. Predictability means the embedding positions can be determined only by the length of embeddable positions and the bit length of watermark.

(3) Robustness. Based on the requirement of U, the watermark algorithm should guarantee that the watermark can be extracted from the DS of embedded data.

3.3. Homomorphic Encryption (HE). Encryption is the most famous method in information security. Homomorphic encryption [18] can translate some operations on plain text into other operations on ciphertext. In our solution, we require that data should be encrypted during transferring and embedding process. For full-HE is slow and costly, we decide to use semi-HE as a compromise that give the consideration to both efficiency and security. We list the requirement of our solution for the semi-HE as follows ($E()$ is encryption function, KEY is the encryption key, and S_i is the target information).

(1) Addition Homomorphism

$$E(KEY, S1 + S2) = E(KEY, S1) \odot E(KEY, S2). \quad (1)$$

(2) Multimap. The absolute value of each of the encryption results depends on the random number it used in different times:

$$E(KEY, S1 : t_1) \neq E(KEY, S1 : t_2). \quad (2)$$

4. Solution Framework

Our solution contains three protocols based on Public-Key Infrastructure (PKI) that is used for distributing public and private key pair combining to each registered ID. The notation used in protocols has been listed at the end of the paper.

4.1. Uploading Stage. In this subsection, we describe the details about uploading stage, including watermark's generating and embedding.

Input: D, L
Output: the threshold of watermark matching μ
(1) **if** $L < 2$ **then**
(2) **return** 100
(3) **end**
(4) **if** *embed* $\{1\}_L$ or $\{0\}_L$ *will change Fea* **then**
(5) **return** 100
(6) **end**
(7) $TESTD = EW(D, \{1 \mid 0\}_L)$
(8) $\mu \longleftarrow 0$
(9) **while** *there is attack method has not been tested* **do**
(10) AD = attacked TESTD
$$\mu = \max(\mu, 100 * \frac{sum(xor(DW(AD), DW(TSETD)))}{L})$$
(11) **end**

Algorithm 1: Generating μ.

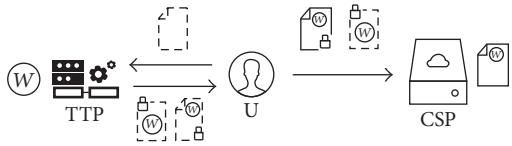


Figure 1: Data flow of updating process.

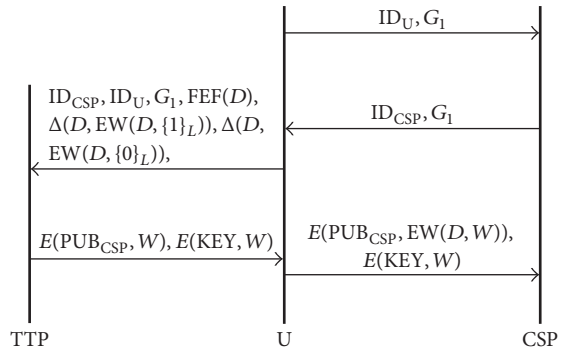


Figure 2: Details of updating process.

Before outsourcing data are transferred to CSP, U needs to embed watermark into his data as shown in Figure 1.

All transferred data are encrypted by CSP's public key or TTP's special key. The details of each process are presented by Figure 2 and introduced in the following steps.

Step 1. U sends CSP his ID and service contract to apply for storing and computing his data. The first contract (G_1) details the responsibilities and obligations of U and CSP and the subalgorithm, including parameters which CSP needs to know, used in the whole solution. The contract is signed by PRI_U to make sure of its integrity. Once CSP does not admit the contract, CSP can reject U's request and the protocol is finished.

Step 2. CSP sends ID_CSP and G_1 to U, which is signed by PRI_CSP. This step means U's request has been permitted. G_1 has been signed twice to make sure that its content has not been changed and will be used as an evidence in the future.

Step 3. U selects a watermark algorithm matching the requirements declared before and threshold μ according to Algorithm 1, which will be signed and attached behind G_1, to embed $\{1\}_L$ and $\{0\}_L$ into his data, where L is the watermark capacity of D and calculates the differences between original data and embedded data as δ_1 and δ_0 according to (3). TESTD is the data embedded with a random sequence of length L, which is used to test the robustness of watermark algorithm; the test round can be done more than 1 time for security purpose. U should make sure that the production of embedding process still belongs to DS(D).

$$\begin{aligned} \delta_1 &= \{\delta_{1i} \mid 1 < i < L\} = \Delta\left(D, E\left(D, \{1\}_L\right)\right) \\ \delta_0 &= \{\delta_{0i} \mid 1 < i < L\} = \Delta\left(D, E\left(D, \{0\}_L\right)\right). \end{aligned} \tag{3}$$

Then, U sends IDs, G_1, FEF(D), δ_1, and δ_0 to TTP for recording and generating watermark W.

Step 4. TTP generates W according to existing data of U that share the same Fea. We present Algorithm 2 as an example for generating watermark here. TTP creates δ_2 by δ_1, δ_0, and W like Algorithm 3.

According to Markov property and predictability, Algorithm 3 guarantees that TTP can create an additive watermark δ_2 based on δ_1 and δ_0. δ_2 is the same as the difference between original data and its copy embedded with W by the selected watermark algorithm. Then, TTP sends the encrypted δ_2 to U as well as signature. Here, we suggest that TTP use two keys to encrypt δ_2. PUB_CSP encrypted copy is for embedding, and TTP's KEY encrypted copy is for

Input: SW(Set of exist watermark of U with same Fea), μ, L(Capacity of File)
Output: A new watermark
(1) $flag = 1$
(2) **while** *flag* **do**
(3) randomly generate a sequence t shorter than $\frac{L}{2}$. **for** *All item x in SW* **do**
(4) **if** $sum(xor(x,t)) < \mu$ **then**
(5) $flag = 0$ break
(6) **end**
(7) **end**
(8) $flag = 1 - flag$
(9) **end**
(10) **return** t

Algorithm 2: Generating W.

Input: δ_1, δ_0, W
Output: δ_2
(1) n← length of W;
(2) $step \leftarrow round(L/n)$;
(3) embedding positions ←random sequence from[1,2,...step] of length $step/2$
(4) **for** $i = 1$ **to** $n + 1$ **do**
(5) **for** $j = 1$ **to** *step* **do**
(6) **if** $2(i-1)step + j > L$ **then**
(7) Break;
(8) **end**
(9) **else**
(10) **if** *j is in embedding positions* **then**
(11) **if** $w_i == 1$ **then**
(12) $\delta_{2((i-1)step+j)} = \delta_{1((i-1)step+j)}$
(13) **end**
(14) **else**
(15) $\delta_{2((i-1)step+j)} = \delta_{0((i-1)step+j)}$
(16) **end**
(17) **end**
(18) **else**
(19) $\delta_{2((i-1)step+j)}$ is equal to $\delta_{(1-w_i)((i-1)step+j)}$
(20) **end**
(21) **end**
(22) **end**
(23) **end**
(24) **return** δ_2

Algorithm 3: Generating $\delta 2$.

erasing in the future which will release the storage burden of TTP.

Step 5. U verifies TTP's signature to make sure that W is valid. Then U uses $\mathrm{PUB_{CSP}}$ to encrypt D and embeds W into D according to the addition homomorphism of encryption algorithm as the following proof has shown, which will get the encrypted file (ED) that contains W.

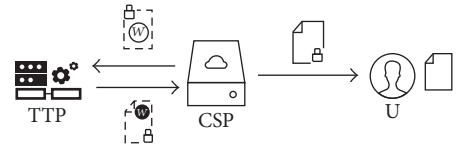


Figure 3: Data flow of downloading process.

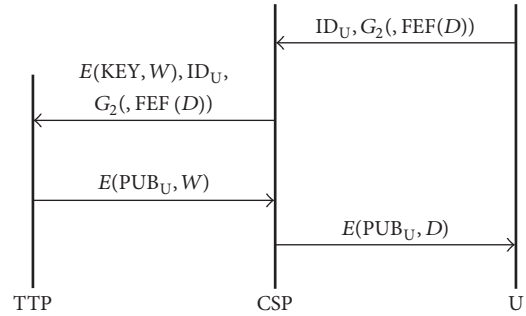


Figure 4: Details of downloading process.

Proof.

$$\begin{aligned}\mathrm{ED} &= E\left(\mathrm{PUB_{CSP}}, \delta_2\right) \odot E\left(\mathrm{PUB_{CSP}}, D\right)\\ &= E\left(\mathrm{PUB_{CSP}}, \delta_2 + D\right)\\ &\because \delta_2 = \Delta\left(D, \mathrm{EW}\left(D, W\right)\right)\\ &\therefore \mathrm{ED} = E\left(\mathrm{PUB_{CSP}}, \Delta\left(D, \mathrm{EW}\left(D, W\right) + D\right)\right)\\ &= E\left(\mathrm{PUB_{CSP}}, \mathrm{EW}\left(D, W\right)\right).\end{aligned} \tag{4}$$

□

U sends δ_2 encrypted by TTP's KEY and ED to CSP. CSP decrypts ED and stores it. Then, the uploading stage is finished.

4.2. Downloading Stage. The downloading stage is much simpler than the uploading protocol, for δ_2 has been stored encrypted in CSP's servers. The data flows are shown in Figure 3.

All data are still encrypted. Details of downloading stage are presented by Figure 4 and introduced as follows.

Step 1. U sends G_2 and $\mathrm{ID_U}$ to CSP. G_2 contains the requirement of retrieving or deletion which need erasing W from ED. U can use FEF(D) to help CSP and TTP search the exact data that he wants.

Step 2. After verifying U's signature, CSP sends G_2 and $\mathrm{ID_U}$ along with encrypted δ_2 to CSP so that CSP can create reversed watermark to erase W from ED.

Step 3. TTP verifies all the information stored in his database. If the information is correct, TTP first decrypts δ_2 and creates the reversed watermark $-\delta_2$. Then TTP encrypts it by $\mathrm{PUB_U}$ and sends it back to CSP.

Step 4. CSP embeds encrypted $-\delta_2$ into ED and then sends it to U. U decrypts receiving file to get his data according to the following proof.

Proof.

$$\begin{aligned} & E\left(\mathrm{PUB}_{\mathrm{U}}, \mathrm{EW}\left(D, W\right)\right) \odot E\left(\mathrm{PUB}_{\mathrm{U}}, -\delta_2\right) \\ & \quad \longrightarrow E\left(\mathrm{PUB}_{\mathrm{U}}, \mathrm{EW}\left(D, W\right) - \delta_2\right) \\ & \quad \longrightarrow E\left(\mathrm{PUB}_{\mathrm{U}}, \mathrm{EW}\left(D, W\right) - \Delta\left(D, \mathrm{EW}\left(D, W\right)\right)\right) \\ & \quad \longrightarrow E\left(\mathrm{PUB}_{\mathrm{U}}, D\right). \end{aligned} \tag{5}$$

□

Once G_2 requires CSP to delete U's data, U cannot download that data in future again, and TTP will create the log of this data and abandon the KEY of δ_2.

4.3. Arbitration Stage. When U finds an unauthorized file L that belongs to DS(D), U can identify the illegal distributor and bring a suit against it.

U should first execute FEF function to get Fea about the leaked data (L) and then provide the L, FEF(L) as well as ID_{U} to TTP. After verifying the information about U, TTP searches the data based on FEF(L) and ID_{U} to get logs of possible leaked data as set S. If S is empty, TTP tells U that this data is not recorded in his database. Otherwise, TTP executes DW(L) according to watermark algorithm and embedding positions of each item of S and calculates the bit error ratio (BER) of DW(L) and W as δ_3. If there is any δ_3 below μ declared in G_1 of that data, TTP believes the CSP signed G_1 violate U's RTBF or privacy. TTP will provide a proof with digital signature to U as a legal evidence.

5. Solution Analysis

Our solution proposed above can solve the problem we mentioned in problem model. The safety of our solution relies critically on the security of subalgorithms like watermarking and encryption algorithm. In this section, we will analyze properties we described in design goals and requirement to each party.

5.1. Effectiveness. Our solution can solve the problem of RTBF as we have mentioned. Once CSP want to violate U's RTBF, he needs to distribute DC(D) to others. If U finds that copy, he can send it to TTP and ask for arbitration, and CSP's crime will be proved. Once U wants to fraud an innocent CSP, U should create a copy that belongs to DS(D) and contains W. However, U has no information about W in plain text. It is technically impossible for CSP to do that if the encryption algorithm is secure enough.

5.2. Security. The security of our solution is based on the fact that U and CSP cannot get information about W. We assume all the subprotocols can satisfy the property we required.

CSP possesses embedded data (ED), δ_2 and $-\delta_2$ encrypted by KEY or $\mathrm{PUB}_{\mathrm{U}}$. CSP wants to create a copy of ED-δ_2, which is impossible if the encryption algorithm is strong enough. Besides, CSP can try to attack ED so that δ_3 are larger than μ. In this case, the robustness of watermark algorithm and FEF function is tested. With the help of μ and FEF, CSP cannot create a useful copy while maintaining the validity of data for distributing.

U possesses D, δ_1 and δ_0 in plain text, δ_2, ED encrypted by $\mathrm{PUB}_{\mathrm{CSP}}$, δ_2 encrypted by KEY. According to the multimap property, U cannot use δ_1 and δ_0 to create δ_2 in polynomial time. Besides, embedding positions will make it harder for both U and CSP to get information about W, though it sacrifices the robustness to some extent. Moreover, considering CSP may leak a part of data and be found by U, U can get a message containing W. U may try to use it to guess other watermarks. The watermark generation is completely random and each watermark shares different length and embedding positions. The possibility of creating a DC to match the watermark is P. Here, we neglect the possibility that a extract watermark can be recognized as two embedded water marks.

$$P \approx \frac{\sum_{i=1}^{\mu L} \binom{L}{i}}{2^L - \sum_{i=1}^{\mu L} \binom{L}{i}}. \tag{6}$$

In conclusion, our solution can make sure that U and CSP cannot get DC of the other one's copy. The robustness of watermark is controlled by U according to the FEF function and watermark algorithm.

5.3. Consumption

U. U outsources local data to CSP for reducing the local data storage space and the cost of complex computing. In our scheme, after uploading data, U can reserve FEF(D) for reducing the cost of searching. U should also do some computation for encrypting and decrypting data.

TTP. TTP has enough storage space for keeping the records of contracts, IDs, Fea, and watermarks for arbitration parts. In this paper, TTP is designed with memory and some necessary computing powers. TTP can take some fee for arbitration requirement so that it will not be annoyed by unsure request and balance the expenses.

CSP. CSP provides large storage space and strong computing power as service. It is reasonable to put the burden of storing outsourced data as well as encrypted δ_2 on CSP.

6. System Design

In this section, we will introduce the experiment we used for verifying the validity and security of our solution. We choose image as U's data to finish our experiment because it is the most popular kind of data used in outsourcing service. Before introducing experiment, we first clarify the subalgorithms we used in our solution.

6.1. Watermark Algorithm. The watermark scheme we used for experiment is Dither Modulation-Quantization Index Modulation (DM-QIM) [20]. It is a classical watermarking scheme and easy for use. Although it has been proved not safe enough [21, 22], it satisfies the requirements we proposed for watermark algorithm.

Generate n = q ∗ p where p and q are both random large prime;
Generate g as a random number of $\mathbb{Z}_n^*$
$\lambda = LCM(p-1, q-1)$
Public key ← (n,g), Private key ← (p,q)
#encryption:
randomly select r < n
$C = g^m * r^n \bmod n^2$
return C
#decryption:
$calculate\lambda$
$m = \dfrac{(c^\lambda \bmod n^2) - 1}{(g^\lambda \bmod n^2) - 1} \bmod n^2$
return m
#addition:
$C_3 = C_1 * C_2 = g^{m1+m2} * (r_1 r_2)^n$
return C_3

Algorithm 4: Paillier.

DM-QIM embeds watermark into transforming domain. It adjusts the value of some coefficients, which is the preknowledge, to embed the message according to (7) where step is the quantizer and d is the dither.

$$\mathrm{EW}(D_i, W_i) = \begin{cases} \mathrm{round}\left(\dfrac{D_i - d}{\mathrm{step}}\right) * \mathrm{step} + d & W_i == 1, \\ \mathrm{round}\left(\dfrac{D_i + d}{\mathrm{step}}\right) * \mathrm{step} - d & W_i == 0. \end{cases} \tag{7}$$

In the extracting process, we use ED_i to reprensent the ouput of $\mathrm{EW}(D_i, W_i)$. According to (8), we can find that different judgments (Jud), which are guessed result before extracting, will lead to different extracting processes and extract different values because of the quantizer. We can add up all $\mathrm{DW}(ED_i)$ that embed same bit of watermark to measure whether Jud is equal to W_i. In any case, we will get the watermark embedded in the picture.

$$\begin{aligned} \mathrm{DW}(ED_i) &= \begin{cases} \mathrm{round}\left(\dfrac{ED_i - d}{\mathrm{step}}\right) * \mathrm{step} + d - ED_i & \mathrm{Jud} == 1,\ W_i == 1, \\ \mathrm{round}\left(\dfrac{ED_i - d}{\mathrm{step}}\right) * \mathrm{step} + d - ED_i & \mathrm{Jud} == 1,\ W_i == 0, \\ \mathrm{round}\left(\dfrac{ED_i + d}{\mathrm{step}}\right) * \mathrm{step} - d - ED_i & \mathrm{Jud} == 0,\ W_i == 1, \\ \mathrm{round}\left(\dfrac{ED_i + d}{\mathrm{step}}\right) * \mathrm{step} - d - ED_i & \mathrm{Jud} == 0,\ W_i == 0, \end{cases} \\ &= \begin{cases} 0 & \mathrm{Jud} == 1,\ W_i == 1, \\ \left(\mathrm{round}\left(\dfrac{ED_i - d}{\mathrm{step}}\right) * \mathrm{step} + d\right) - ED_i & \mathrm{Jud} == 1,\ W_i == 0, \\ \left(\mathrm{round}\left(\dfrac{ED_i + d}{\mathrm{step}}\right) * \mathrm{step} - d\right) - ED_i & \mathrm{Jud} == 0,\ W_i == 1, \\ 0 & \mathrm{Jud} == 0,\ W_i == 0. \end{cases} \end{aligned} \tag{8}$$

This watermark scheme embeds 1 or 0 into each selected coefficient as Algorithm 3 which means it satisfies the Markov property. In our solution, we split image into several 8 × 8 nonoverlapping blocks firstly and use DCT to transform these blocks into transform domain, which means all the coefficients can be placed in a meaningful place so that TTP can determine which position to embed. Thus, this scheme has predictability.

In our solution, DCT coefficients below 0.4 are chosen for watermarks. Embedding positions are selected according to the value of $(i + j) \bmod 2$, where i and j are the coordinates of the coefficient. In our demo, the step is 100 and the dither is 25.

6.2. Encryption. In our system, we use AES and Paillier [23] as encryption algorithm that can fit solution's requirements. AES is a famous symmetric encryption algorithm [24] which is fast and safe. Paillier is a semi-HE that supports additive operations in ciphertext according to Algorithm 4 where $L(x) = (x - 1)/n$.

Although Paillier allows user to do addition, negative numbers and decimals are not allowed to calculate. Because δ_2 and $-\delta_2$ always need to encrypt negative numbers and decimals, we suggest U and CSP do as shown in Algorithms 5 and 6 which can solve this problem.

Besides, δ_0 and δ_1 may leak some information about the image; we recommend that U adds a mark to δ_0 and δ_1 which can be subtracted after decrypting δ_2.

6.3. Perceptual Hash Algorithm (PHA). We choose PHA as FEF function for it can reflect the content about image in its low frequency coefficients which is also used in searching engine [25]. Algorithm 7 shows pHash we used as FEF function.

(a) (b) (c)

Figure 5: The retrieved picture (a), embedded picture (b), and the original picture (c).

Input: message m, amp (integer amplifier which decides the accuracy of data)
Output: pretreated m, balance
(1) $m_1 = round(m * amp)$
(2) **if** m_1 *contains negative* **then**
(3) balance = round($\frac{\max(-\min(m_1), \max(m_1))}{10}$) $*$ 100
(4) **end**
(5) **else**
(6) balance = round($\frac{\max(m_1)}{10}$) $*$ 100
(7) **end**
(8) **for** $i = 1$ **to** L **do**
(9) $m_{2i} = m_{1i} + balance$
(10) **end**
(11) **return** m_2, *balance, amp*

Algorithm 5: Pretreatment: before encryption.

Input: decrypted dm, amp, balances
Output: message m
(1) FB=sum(balances)
(2) **for** $i = 1$ **to** L **do**
(3) $m_{1i} = dm_i - FB$
(4) **end**
(5) $m = round(m_1/amp)$ **return** m

Algorithm 6: Pretreatment: after encryption.

Input: I
Output: hash
(1) h_r = Resize(I, [32, 32])
(2) h_t=DCT(h_r)
(3) sum=0
(4) **for** $i, j = 1$ **to** 8 **do**
(5) $h_{lij} = h_{tij}$
(6) $sum = sum + h_{tij}$
(7) **end**
(8) mid=$\frac{sum}{64}$
(9) **for** $i, j = 1$ **to** 8 **do**
(10) $h_{lij} = sgn(h_{lij} - mid)$
(11) **end**
(12) **return** h_l

Algorithm 7: PHA.

7. Tolerance about System

We first evaluate the tolerance about system. We assume U that has uploaded a large number of images to CSP that have been registered in TTP. One of his images, which has been required to delete, is attractive that CSP wants to distribute it for benefits. CSP needs to erase the watermark embedded in image while he knows nothing about the watermarks' algorithm. So CSP could only use some basic function to attack it. Geometric attacks are not in considered for user that can get the information about watermark by recovering it in most cases, which is dangerous for CSP because U can use this information to create a copy of ED easily. We will consider three types of attack means: JPEG compression (JC), Gaussian filter (GF), and White Gaussian Noise (WGN) to represent the loss compression, filter, and noise attack in the following parts.

We use peak signal-to-noise ratio (PSNR) and bit error ratio (BER) as two indicators that evaluate the performance about our solution. In this section, we will evaluate DC of watermarked Lena provided by each attack mean of different parameters. To compare, the retrieved image's (Figure 5) PSNR maintains 313 dB in our solution.

7.1. JPEG Compression Test. JPEG compression is one of the most popular compression ways that is used for maintaining the main information in smaller size. We want to examine whether CSP can distribute a compressed version data illegally.

Table 1: Watermark's tolerance to GF.

Scale	σ 0.1	0.2	0.3	0.5	0.8	1	2	4	8	10
	BER									
2	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11
4	0.11	0.11	0.11	0.11	0.01	0.01	0.32	0.46	0.48	0.48
8	0.12	0.12	0.12	0.12	0.01	0.01	0.28	0.46	0.48	0.48
10	0.12	0.12	0.12	0.12	0.01	0	0.28	0.46	0.48	0.48

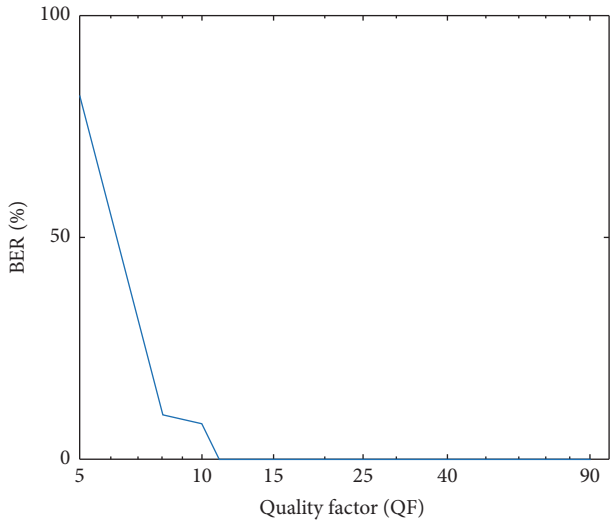


Figure 6: Watermark tolerance to JPEG compression.

Figure 6 shows that the BER decreased rapidly as the quality factor (QF) grows. When QF is 5, which is not a normal choice for compression, Fea of the attacked picture (Figure 8) has changed. This means that our solution can be against the JPEG compression if $\mu > 10\%$.

7.2. Gauss Filter. Filter is the riskiest attack for DM-QIM, since it erases the details within each block of selected scale by adjusting DCT coefficients. As σ grows, picture will become more and more smooth. The mid one in Figure 8 is attacked by GF with scale = 8 and $\sigma = 2$. It suggests that PHA we have used is not the best way to represent the content of image.

Table 1 and Figure 7 show the PSNR and BER affected by GF in different scales and σ. We can notice that GF with scale of 2 has no risk to our solution. When σ is close to 1, BER of attacked image decreases to nearly 0 and the PSNR grows. We consider this as a kind of tolerance to GF. As σ grows continually, BER grows rapidly and the watermark and the detail of picture are erased.

To be against these attacks means that we can change the watermark algorithm or amplify the step as well as dither, which will introduce more noise to embedded picture. This is completely a trade-off between security and the validity of data. The restriction to μ in this experiment is 30%.

7.3. White Gaussian Noise (WGN). Noise is another kind of attack, which will quickly decrease PSNR of image. We use

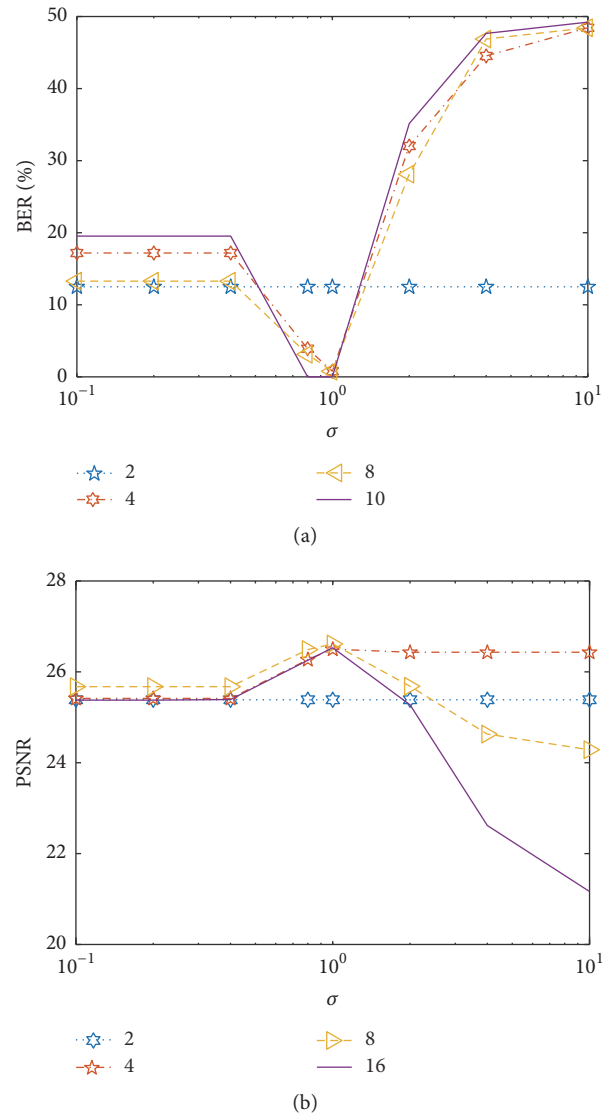


Figure 7: BER (a) and PSNR (b) change according to the watermark.

Gaussian noise to attack our picture. GF and compression will erase the details of images. This will help the attacker decrease the noise watermark introduced in and raise the PSNR of picture in some degree. However, WGN introduces more noise into picture (Figure 8) to cover the watermark which will decrease PSNR quickly and change the Fea.

(a) (b) (c)

Figure 8: Pictures attacked by JC (a), GF (b), and WGN (c).

Table 2: Watermark's tolerance to WGN.

PSNR	23.35	17.94	16.8	16.24	15.88	14.69	13.64	12.93
BER	0	0	0.09	0.26	0.39	0.51	0.52	0.52

Table 2 shows the change of BER according to WGN. The PSNR of Figure 8 is 15.88 which suggests that μ should be at least 26% so that they can defend WGN to some extent in our solution.

8. Capacity of System

In this section, we will take the arbitration stage into consideration. We assume that U finds a picture Y which may be a DC of his deleted data. For U may have not backed up his data, U uploads Y, FEF(Y), and ID_{U} to TTP so that TTP can determine which CSP may leak his data. In this experiment, we will test the capacity of our solution, which is the number of watermarks that are embedded into user's data with same Fea. There are several ways to generate a secure watermark [26]. For simplicity, we choose Algorithm 8 to generate our watermark easily. We named the result of Algorithm 8 as fixed Hamming distance codes, the Hamming distance of each element within answer is no less than the input limits μ. Fixed Hamming distance code allows us to identify the log about data as well as its contracts. And the watermark will be able to defend the attacks as long as U has to test μ according to Algorithm 1.

The final capacity of a TTP will be calculated by (9) where cap(D) means the span of D's value.

$$N = \left(\sum_{i=L_{\min}}^{L_{\max}} \mathrm{Len}\left(\mathrm{watermarks}^{i}\right) \right) * \mathrm{cap}\,(\mathrm{FEA}) * \mathrm{cap}\,(\mathrm{ID})\,. \tag{9}$$

8.1. Result. We first use 30% as threshold, 10 seconds as time limit and 128 bits as the length of watermark. We get at least 1000 watermarks. we select the 500th watermark as the embedded watermark and do attacks as Table 3 presents.

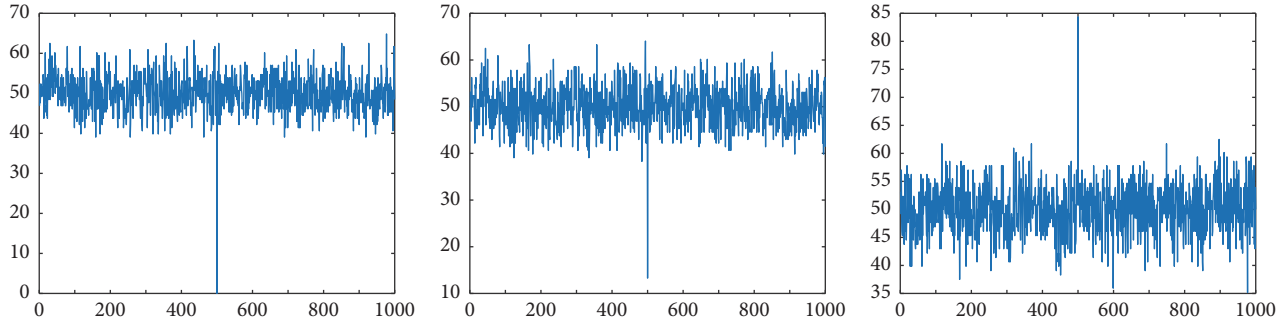
Input: μ, L
Output: $watermarks^L$
(1) $watermarks^L$ = empty set
(2) threshold=μL
(3) flag=1
(4) **while** *not reach time-limit* **do**
(5) random generate a temp watermark t of length L
(6) flag=1
(7) **for** each w in $watermarks^L$ **do**
(8) **if** *sum(XOR(w, t)<threshold)* **then**
(9) flag=0
(10) break
(11) **end**
(12) **end**
(13) **if** *flag* **then**
(14) Add t into $watermarks^L$
(15) **end**
(16) **end**
(17) **return** $watermarks^L$

Algorithm 8: Fixed Hamming distance coding.

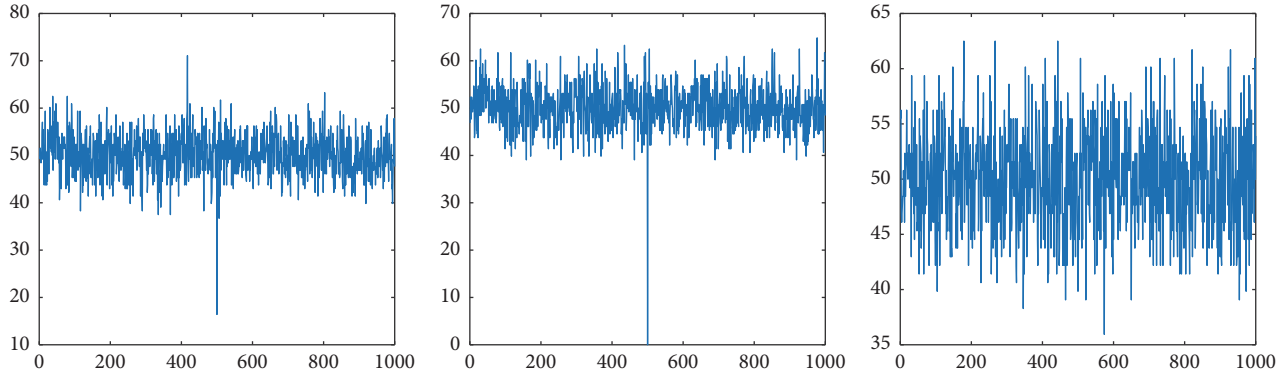
The results are presented by Figure 9 (The y-axis represents BER and x-axis represents sequence number of images). It suggests that our protocol can identify the certain data of it within our database and charge the CSP successfully under the predicted attacks. Fixed Hamming distance code makes sure that the robustness of this protocol is only determined by watermark algorithm and encryption method. The third picture in each line of Figure 9 shows that if the picture is overattacked, we cannot determine the source of the picture from watermark.

Table 3: Test attacks.

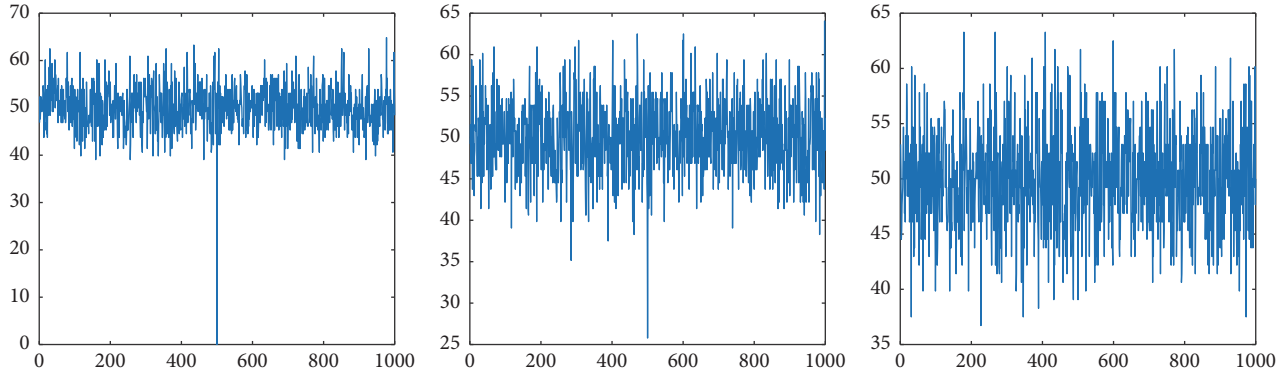
JPEG compression	QF = 25	QF = 8	QF = 5
GF	sigma = 0.3	sigma = 1	sigma = 4
WGN	PSNR = 18	PSNR = 16.75	PSNR = 16.41



(a) JPEG compression for QF of 25, 8, and 5



(b) GF for sigma of 0.3, 1, and 4



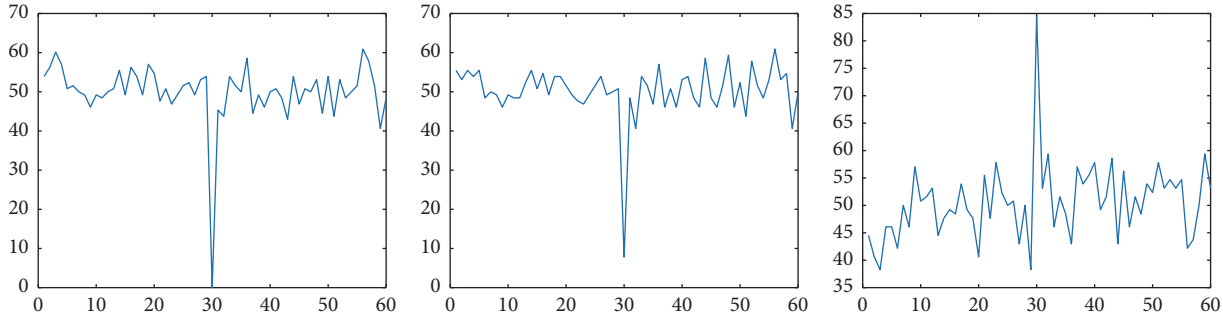
(c) WGN for PSNR of 18, 16.75, and 16.41

Figure 9: Test for $\mu = 0.3$.

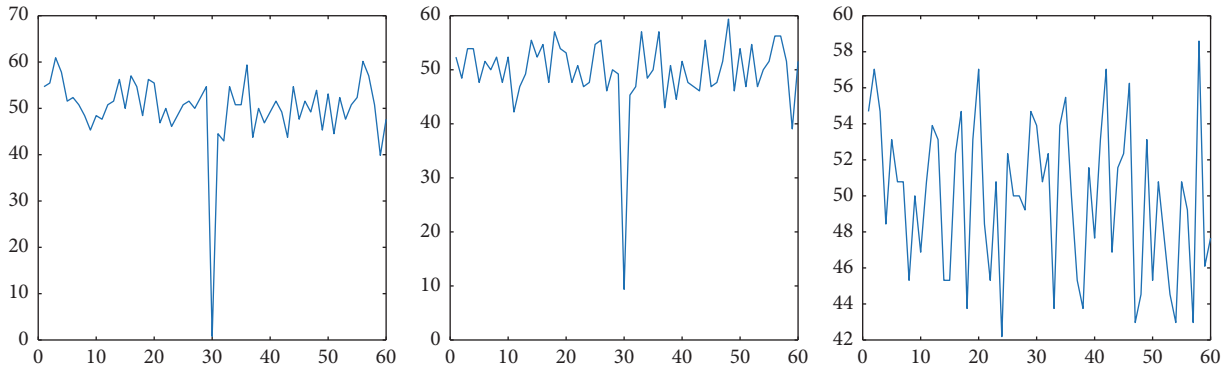
In addition, we raise the threshold to 40% which results in a quickly decreasing of capacity. We can only get 60 watermarks within 10 seconds. We select the 30th watermark for embedding and do the same tests. The results are shown in Figure 10 (The y-axis represents BER and x-axis represents sequence number of images). It suggested that raising up threshold is not economic to increase the robustness of watermark algorithm for it decreases the number of watermarks largely.

9. Conclusion

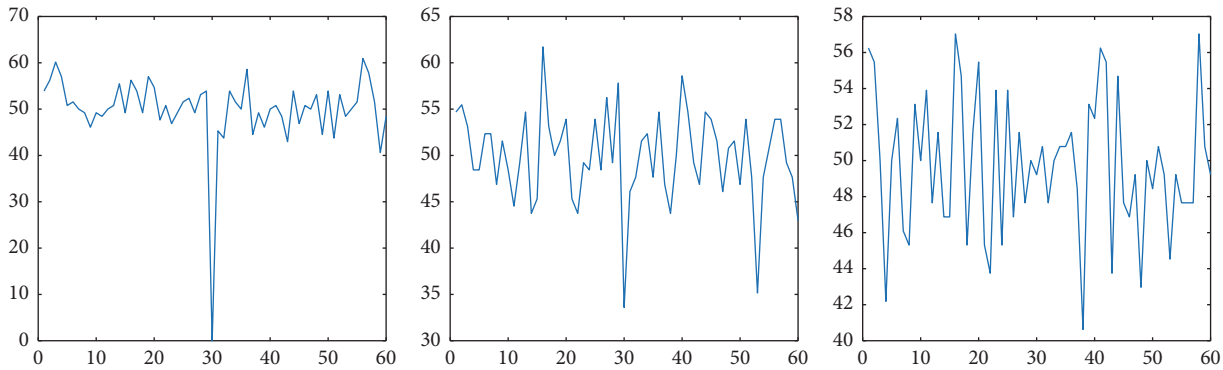
In this paper, we propose a Cloud-User protocol as a solution to solve the Right to Be Forgotten problem technically. Our solution supports confirmed deletion of plain data that is stored in CSP's servers. To achieve security goals, our solution combines the existing homomorphic cryptography, watermark techniques, minimum Hamming coding, and the content-based feature extraction so that the innocent party



(a) JPEG compression for QF of 25, 8, and 5



(b) GF for sigma of 0.3, 1, and 4



(c) WGN for PSNR of 18, 16.75, and 16.41

Figure 10: Test for $\mu = 0.4$.

will not suffer losses by the other one's attack. We implement a prototype of our solution to demonstrate its availability and practicality.

10. Future Work

For future work, there are still some aspects worth thinking. Firstly, the algorithms we used in prototype are not the best ones that fit our solution. Choosing a better encryption algorithm and watermark scheme may decrease the cost of communication and computation for U and TTP.

Secondly, a better FEF can help protecting U's right and raise the robustness of our protocol. We treat the combination of FEF and watermark scheme as the most challenging question for our solution.

Thirdly, every time user retrieves his data will cost a lot for all three parties. Designing a better drawing back protocol can raise the efficiency of our solution.

Finally, as a large pile of data is plain text in CSP's server, how to provide preview of data base on its content like existing systems [27] in low cost while not leaking the information of watermarks is waiting to be solved.

Notations

CSP: Cloud Service Provider
U: User
TTP: Trusted Third Party
D: Data
W: Digital watermark

DC(D):	Derivative copy of D
DS(D):	Derivative set of D
DW(D):	Watermark extraction function that extracts watermark from D
EW(D, M):	Watermark embedding function that embeds M into D
FEF(D):	Feature of D
G_i:	The ith round contract
ID_x:	Identity of x
KEY:	Key for symmetric cryptography
PUB_x:	Public key of x
PRI_x:	Private key of x
a_n:	A string generated by combination of a that has length n
D(KEY/PUB, D/M):	Decryption function for D/M while key is KEY/PUB
E(KEY/PUB, D/M):	Encryption function for D/M while key is KEY/PUB
$\Delta(D_1, D_2)$:	The difference between D_1 and D_2 as $\{D_{1i} - D_{2i} \mid 1 < i < \mathrm{Len}(D_1)\}$.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported in part by the Natural Science Foundation of China under Grants U1636201 and 61572452.

References

[1] Directive, EU,95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data,Official Journal of the EC, 23(6), 1995.

[2] Proposal for a Regulation of the European Parliament and of the Council on the Protection of Individuals with Regard to the Processing of Personal Data and on the Free movement of Such Data (General Data Protection Regulation), 2012.

[3] Steinberg, An act to add Chapter 22.1 (commencing with Section 22580) to Division 8 of the Business and Professions Code, relating to the Internet,California senate, 2013.

[4] Google, *European privacy requests for search removals*, 2017, https://www.google.com/transparencyreport/removals/europeprivacy.

[5] Z. Xia, X. Wang, L. Zhang, Z. Qin, X. Sun, and K. Ren, "A privacy-preserving and copy-deterrence content-based image retrieval scheme in cloud computing," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 11, pp. 2594–2608, 2016.

[6] Y. Tang, P. P. C. Lee, J. C. S. Lui, and R. Perlman, "Secure overlay cloud storage with access control and assured deletion," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 6, pp. 903–916, 2012.

[7] R. Geambasu, T. Kohno, A. A. Levy et al., "Vanish: Increasing Data Privacy with Self-Destructing Data," in *Proceedings of the USENIX Security Symposium*, pp. 299–316, 2009.

[8] G. Roxana, K. Tadayoshi, A. Amit et al., "Increasing data privacy with self-destructing data," in *Proceedings of the USENIX Security09*, pp. 299–316, Berkeley, CA, USA, 2009.

[9] C. Gentry, "Fully Homomorphic Encryption Using Ideal Lattices," in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC '09*, pp. 169–178, June 2009.

[10] R. Rivest L, L. Adleman, and L. Dertouzos M, "On data banks and privacy homomorphisms," *Foundations of Secure Computation*, vol. 4, no. 11, pp. 169–180, 1978.

[11] Z. Fu and X. Cao, "An Outsourcing Data Storage Scheme Supporting Privacy Preserving and Data Hiding Based on Digital Watermarking," in *Proceedings of the International Conference on Cloud Computing and Security*, pp. 468–474, Springer, 2016.

[12] J. Long, D. Zhang, C. Zuo, J. Duan, and W. Huang, "A robust low-overheadwatermarking for field authentication of intellectual property cores," *Computer Science and Information Systems*, vol. 13, no. 2, pp. 609–622, 2016.

[13] T. Bianchi and A. Piva, "Secure watermarking for multimedia content protection: a review of its benefits and open issues," *IEEE Signal Processing Magazine*, vol. 30, no. 2, pp. 87–96, 2013.

[14] N. Memon and P. W. Wong, "A buyer-seller watermarking protocol," *IEEE Transactions on Image Processing*, vol. 10, no. 4, pp. 643–649, 2001.

[15] C.-L. Lei, P.-L. Yu, P.-L. Tsai, and M.-H. Chan, "An efficient and anonymous buyer-seller watermarking protocol," *IEEE Transactions on Image Processing*, vol. 13, no. 12, pp. 1618–1626, 2004.

[16] B. Chen and G. W. Wornell, "Quantization index modulation: a class of provably good methods for digital watermarking and information embedding," *Institute of Electrical and Electronics Engineers Transactions on Information Theory*, vol. 47, no. 4, pp. 1423–1443, 2001.

[17] P. Tao and A. M. Eskicioglu, "A robust multiple watermarking scheme in the Discrete Wavelet Transform domain," in *Proceedings of the Internet Multimedia Management Systems V*, pp. 133–144, October 2004.

[18] J. Molina-Garcia, R. Reyes-Reyes, V. Ponomaryov, and C. Cruz-Ramos, "Watermarking algorithm for authentication and self-recovery of tampered images using DWT," in *Proceedings of the 9th International Kharkiv Symposium on Physics and Engineering of Microwaves, Millimeter and Submillimeter Waves, MSMW 2016*, pp. 1–4, June 2016.

[19] O. Wahballa, A. Abdalla, K. Hamdnaalla, M. Ramadan, and C. Xu, "An efficient and secure certificateless public key watermarking scheme based on 1VD-DWT," in *Proceedings of the 2016 IEEE International Conference on Cloud Computing and Big Data Analysis, ICCCBDA 2016*, pp. 183–188, July 2016.

[20] B. Chen and G. Wornell, "Digital watermarking and information embedding using dither modulation," in *Proceedings of the 1998 IEEE Second Workshop on Multimedia Signal Processing*, pp. 273–278, Redondo Beach, CA, USA.

[21] P. Bas and J. Hurri, "Security of DM quantization watermarking schemes: a practical study for digital images," in *Proceedings of the International Workshop on Digital Watermarking*, pp. 186–200, Springer, 2005.

[22] B. Matam and D. Lowe, "Watermark-only security attack on DM-QIM watermarking: Vulnerability to guided key guessing," in *Proceedings of the Crime Prevention Technologies and Applications for Advancing Criminal Investigation*, pp. 85–106, IGI Global, 2012.

[23] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proceedings of the International*

Conference on the Theory and Applications of Cryptographic Techniques, pp. 223–238, 1999.

[24] C.-C. Lu and S.-Y. Tseng, "Integrated design of AES (Advanced Encryption Standard) encrypter and decrypter," in *Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors, ASAP 2002*, pp. 277–285, July 2002.

[25] E. Klinger, "pHash The open source perceptual hash library," 2017, http://www.phash.org/.

[26] D. Boneh and J. Shaw, "Collusion-secure fingerprinting for digital data," *Institute of Electrical and Electronics Engineers Transactions on Information Theory*, vol. 44, no. 5, pp. 1897–1905, 1998.

[27] S. Pandey, P. Khanna, and H. Yokota, "A semantics and image retrieval system for hierarchical image databases," *Information Processing & Management*, vol. 52, no. 4, pp. 571–591, 2016.