Journal of Information Hiding and Multimedia Signal Processing ©2015 ISSN 2073-4212
Ubiquitous International Volume **6**, Number **2**, March **2015**

A Robust Image Encryption Scheme Based on RSA and Secret Sharing for Cloud Storage Systems

Hsiao-Ling Wu [1] and Chin-Chen Chang [1,2]

[1]Department of Information Engineering and Computer Science
Feng Chia University
No. 100, Wen-Hwa Rd., Taichung, 40724 Taiwan
wuhsiaoling590@gmail.com
[2]Department of Computer Science and Information Engineeringe
Asia University
No. 500, Lioufeng Rd., Wufeng, Taichung, 41354 Taiwan
*corresponding author: alan3c@gmail.com

Received June, 2014; revised September, 2014

Abstract. *To date, the cloud storage system has been well developed. It is a system that provides users with large storage, high computation ability, and convenience. However, users cannot fully trust a cloud storage system because it is an "honest but curious"server. For this reason, protecting data confidentiality and data integrity are two important issues in a cloud storage system. In this paper, we present our design a secure scheme based on RSA encryption and Shamirs secret sharing schemes. The experimental results and security analysis show that our scheme can effectively ensure data confidentiality and check data integrity. In addition, we compare the computational cost with RSA encryption scheme.*

Keywords: Cloud storage system, Data confidentiality, Data integrity, RSA cryptosystems, Secret sharing

1. **Introduction.** With the rapid development of the Internet and digital technology, people own a lot of "large-data,"such as images and videos. In the past, people stored this data in a personal computer or USB flash drive. When large-data is stored in a personal computer, it cannot be backed up easily, and storing extensive data on a USB flash drive is costly and can be risky. In a worst-case scenario, large-data stored in a personal computer or USB flash drive might be lost, or the storing device might break. Therefore, portability and data that are easy to backup are basic requirements for data storage. Cloud technology has undergone significant development recently, including cloud storage, which is an importation service of cloud technology. A cloud storage system provides certain advantages for users, including large storage, high computation ability, and convenience. Users can pay a relatively low price to obtain the same quality of service. Hence, cloud storage service is widely applied to the real world.

For instance, Samsung's the smart camera products are an application example of the cloud storage system. At first, the user uses a smart camera to take a picture. Then the user opens the Wi-Fi function on the smart camera. Consequently, the user can directly upload the picture from the smart camera to the cloud storage system via Wi-Fi. In this way, the user does not worry about low memory or memory loss problems. Admittedly, a cloud storage system provides many advantages to users, but some security misgivings will be generated. The main cause of security misgivings is that the cloud storage provider is an “honest but curious”server. The content of stored data is obtained without an encryption procedure. Furthermore, the user’s personal privacy is divulged. Therefore, it is important to ensure data privacy and integrity in a cloud storage system. The literature has proposed many schemes to ensure the security of cloud systems [1-7].

The most common method of ensuring data confidentiality is to encrypt the data before the data are uploaded to the cloud storage system [8-13]. When a user wants to upload data to the cloud system, she/he uses a public key or secret key to encrypt the data. Then the encrypted data are transferred to the cloud system. Once the user wants to obtain the data, she/he downloads the encrypted data and uses the private key or secret key to decrypt it. In this manner, data confidentiality can be protected. However, the user computes at least one-time encryption and one-time decryption processes. If a user needs to back up copy to multiple cloud systems, computation costs will increase significantly.

Some existing schemes [14-17] utilize an independent auditing service to ensure data integrity. When data are stored in a cloud storage system, a user may worry that the data will be lost or damaged. Hence, a user can request an auditing service to check the datas integrity. In an independent auditing service, neither the user nor the cloud system provider is the auditor. In order to provide unbiased auditing, the auditor should be a third party. Therefore, there are two assumptions in this situation. The first is that the third-party auditor can be trusted. The other is that the third-party auditor cannot know the content of the data.

According to the aforementioned assumptions, the objective of our proposed scheme was to achieve security and high efficiency for images stored in multiple cloud storage systems. The contributions of our proposed scheme are listed below:
(1) We utilize the concepts of Shamirs secret sharing [18] and the RSA encryption cryptosystem [19] to design our scheme.
(2) In our scheme, an authorized user can check data integrity by her/him-self without an unbiased auditor. Even if some shadows stored in multiple cloud storage systems are lost or broken, the user can still check data integrity.
(3) Our scheme satisfies two basic security requirements: data confidentiality and data integrity. Furthermore, our scheme also can resist statistical and entropy-based attacks, which makes it highly robust.
(4) Our scheme is more efficient than the normal RSA encryption scheme. More specifically, the user needs to perform a one-time encryption operation and a one-time decryption operation in the normal RSA encryption scheme. However, in our proposed scheme, the user performs Shamirs secret sharing procedure and decryption operation only once.

The remainder of this paper is organized as follows. In Section 2, we introduce Shamir’s secret sharing scheme and an RSA encryption scheme. Then, in Section3, we propose a secure encryption scheme for cloud storage systems. The experimental results and security analysis are presented in Sections 4 and 5. Finally, Section 6 presents the conclusions.

2. **Related works.** In this section, we briefly review the basic fundamentals of our scheme. In Subsection 2.1, we introduce Shamir’s secret sharing scheme [18] and in Subsection 2.2, we introduce the RSA encryption scheme [19].

2.1. **Review of Shamir’s secret sharing scheme.** In 1979, Shamir proposed a (t, n) threshold secret sharing scheme [18]. In this sharing scheme, the secret data S is divided into n shares $S_1, S_2, \ldots, S_n$ and the goals are shown below:
(1) When anyone collects t or more than t shares, the secret data S can be easily reconstructed.
(2) When anyone collects $t-1$ or fewer than $t-1$ shares, the secret data S cannot be

reconstructed.
Here, we briefly show how to divide secret data S into n shares in the share-generation phase, and how to reconstruct the secret data S in the secret-reconstruction phase.
A. Share-generation phase. At first, we randomly choose $t-1$ random numbers, a_1, a_2, ..., a_{t-1}, and generate a polynomial function $f(x) = S + a_1 x + a_2 x^2 + \ldots + a_{t-1} x^{t-1} \bmod P$, where $P>S$ and a_1, a_2, ..., a_{t-1} belong to a finite field $GF(P)$. Then, we compute n shares y_i as $f(x_i)$, where x_i means public information. Finally, y_i is securely distributed to shareholder U_i via a secure channel.
B. Share-reconstruction phase. Once we collect at least t shares from shareholders, we can easily reconstruct the original polynomial function $f(x)$ by using the following Lagrange interpolating formula:

$$f(x) = \sum_{i=1}^{t} y_i \prod_{j=1, j \neq i}^{t} \frac{x - x_j}{x_i - x_j}. \tag{1}$$

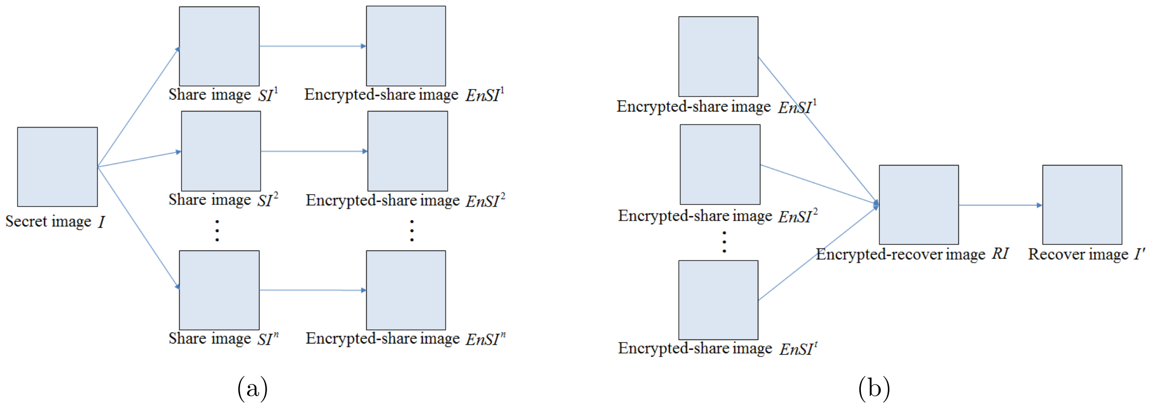
The secret data S is equal to $f(0)$.

2.2. **Review of RSA encryption scheme.** In 1978, Rivest, Shamir and Adleman proposed a public key cryptosystem [19]. Their scheme is divided into two phases: the key generation phase and the encryption/decryption phase. Here, we assume that Alice performs the key generation to obtain her public and private keys, and Bob wants to send a message to Alice in the encryption/decryption phase.
A. Key generation phase. At first, Alice chooses two strong primes [20, 21], p and q, such that $p = 2p' + 1$ and $q = 2q' + 1$, where p' and q' are two large primes. Alice continues to choose a public key e, such that $\gcd(e, \phi(n)) = 1$, where $N = p \times q$ and $\phi(n) = (p-1) \times (q-1)$. The details of Euler's totient function $\phi(n)$ can be found in [22]. When Alice obtains the public key e, she can compute the corresponding private key d, such that $ed \equiv 1 \pmod{\phi(n)}$. Finally, Alice publishes the public key (e, N), and keeps the private key (d, p, q) secretly.
B. Encryption/decryption phase. When Bob wants to send a message M to Alice, he must use Alice's public key to encrypt message M as $C = M^e \mod N$. Then Bob sends the ciphertext C to Alice via an insecure channel. Once Alice receives the ciphertext C, she can obtain message M by using the private key to decrypt C, i.e. $M = C^d \mod N$.

3. Proposed Scheme.

In this section, we propose a novel scheme to save a secret image in multiple cloud storage systems with confidentiality, robustness, and integrity. First, we outline the architecture of our scheme; then, we describe our scheme in detail.

3.1. **Architecture of our scheme.** Our scheme features two phases, 1) the construction phase, and 2) the recovery phase. In the construction phase, the user uses Shamir's secret sharing scheme to share an original secret image I into several share images SI^k, and uploads SI^k to the cloud storage system $Cloud^k$, where $k = 1, 2, \ldots, n$. When each cloud storage system $Cloud^k$ receives share image SI^k, the cloud storage system $Cloud^k$ can encrypt the share image SI^k into the encrypted-share image $EnSI^k$ by using the user's public key. The encrypted-share image $EnSI^k$ is the cipher image. Finally, the encrypted-share image $EnSI^k$ is stored on the database of the cloud storage system $Cloud^k$, where $k = 1, 2, \ldots, n$. Figure 1(a) illustrates the flowchart of the construction phase. In the recovery phase, the user downloads the cipher image from multiple cloud storage systems, and uses the interpolating formula to recover the image. In this paper, we call this image an encrypted-recover image RI. Then the user uses the private key to decrypt the encrypted-recover image into the recover image I'. If the private key is corrected, recover image I' will be equal to the original image I. Figure 1(b) illustrates the flowchart of the recovery phase. The phases are discussed in detail in Subsections 3.2 and 3.3, respectively.


Secret image I
Share image SI^1
Share image SI^2
Share image SI^n
Encrypted-share image $EnSI^1$
Encrypted-share image $EnSI^2$
Encrypted-share image $EnSI^n$
Encrypted-share image $EnSI^1$
Encrypted-share image $EnSI^2$
Encrypted-share image $EnSI^t$
Encrypted-recover image RI
Recover image I'

(a) (b)

FIGURE 1. Flowchart of the construction phase and the recovery phase

3.2. **Construction phase.** In this phase (Fig. 1(a)), user U needs to generate n share images SI^k for secret image I, and each cloud storage system $Cloud^k$ will encrypt the share images SI^k for user U, where $k = 1, 2, \ldots, n$. Here, we assume that user U already generates the public and private keys according to Subsection 2.1, and secret image I is a gray image with a size of $H \times W$ pixels. User U and the cloud storage systems execute the following steps to complete this phase.

Step C1. For each pixel value $I(i,j)$ of secret image I, one polynomial function will be generated by user U. First, user U needs to choose t random numbers, $a_{i,j}^1, a_{i,j}^2, \ldots, a_{i,j}^t$, and compute a polynomial function $f_{i,j}(x) = I(i,j) + a_{i,j}^1 x + a_{i,j}^2 x^2 + \ldots + a_{i,j}^t x^t \mod P$, where $1 \leq i \leq H$, $1 \leq j \leq W$, and $I(i,j)<P$. Therefore, for secret image I, there are $H \times W$ polynomial functions to be obtained.

Step C2. Then user U chooses n random numbers, $x_1, x_2, \ldots, x_n$. Subsequently, random number x_k is chosen as an input, and the computed value, $g^{f_{(i,j)}(x_k)}$ is assigned to the pixel value $SI^k(i,j)$ of the share images SI^k, where $1 \leq k \leq n$, $1 \leq i \leq H$, $1 \leq j \leq W$, and g is a generator of Z_N^*. When n share images are generated, user U will upload share image SI^k into the cloud storage system $Cloud^k$, respectively.

Step C3. Upon receiving share image SI^k from user U, the cloud storage system $Cloud^k$ will employ the user's public key to compute the encrypted-share image $EnSI^k$ as $EnSI^k = (SI^k)^e \mod N$. After that, the encrypted-share image $EnSI^k$ will be stored in the database of the cloud storage system $Cloud^k$.

3.3. **Recovery phase.** In this phase (Fig. 1(b)), user U can collect the t encrypted-share images to get the encrypted-recover image RI, and use the self-private key to recover the original secret image I that he/she wants to recover.

Step R1. First, user U downloads t encrypted-share images $EnSI^k$ from the t cloud storage systems. Second, user U can obtain the encrypted-recover image RI according to the following formula:

$$RI(i,j) = \prod_{k=1}^{t} EnSI^k(i,j)^{\prod_{u \neq k} \frac{-x_u}{x_k - x_u}}, \tag{2}$$

for $1 \leq i \leq H$ and $1 \leq j \leq W$.

Step R2. Upon obtaining the encrypted-recover image RI, user U computes the recover image I' as $I'(i,j) = \log_g RI(i,j)^d$ for $1 \le i \le H$ and $1 \le j \le W$. If the private key d is correct, the recover image I' is equal to the original secret image I; otherwise, I' and I are not the same.

4. **Experimental results.** In this paper, the four gray-scale images (512 × 512 pixels) are used in our scheme, i.e., Lena, Baboon, F-16, and Peppers. Here, the four gray-scale images were like the secret images shown in Figure 2. In the construction phase, Shamir's (3, 4) threshold secret sharing and the RSA cryptosystem are considered. Therefore, the share images and the encrypted-share images of Lena are shown in Figures 3 and 4. When user U performs the recovery phase, he/she can obtain the encrypted-recover and the recover images shown in Figures 5 and 6. Fig. 2(a), Fig. 2(b), Fig. 2(c), and Fig. 2(d) are same as Fig. 6(a), Fig. 6(b), Fig. 6(c), and Fig. 6(d), respectively. Experimental results demonstrate that our scheme can retrieve the original secret image without any lossless.

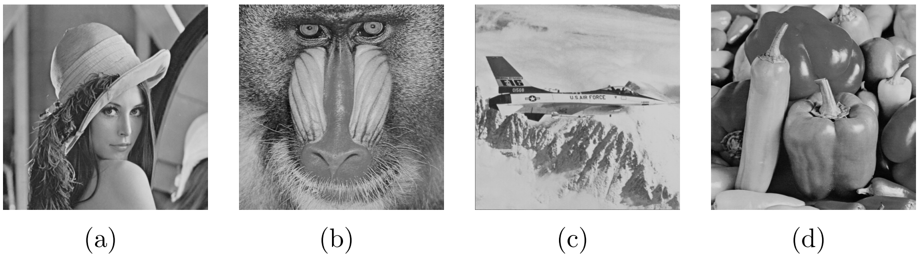


FIGURE 2. The secrete images : (a) Lena, (b) Baboon, (c) F-16, and (d) Peppers

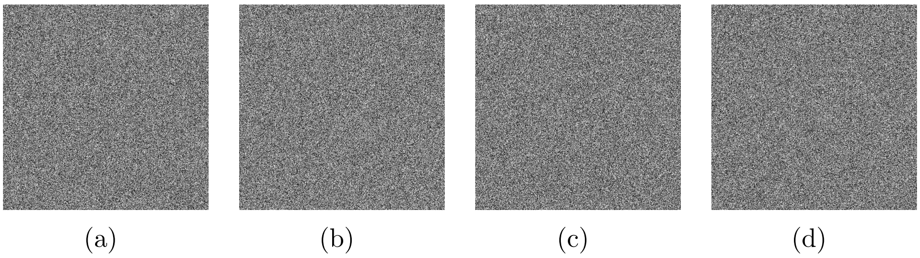


FIGURE 3. Experimental results of the share images of Lena : (a) share image 1, (b) share image 2, (c) share image 3, and (d) share image 4

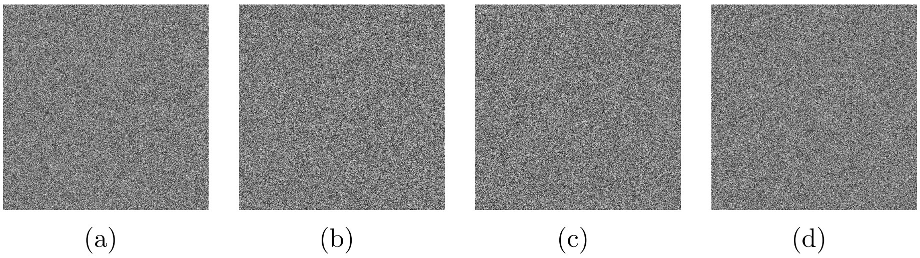


FIGURE 4. Experimental results of the encrypted-share images of Lena (a) encrypted-share image 1, (b) encrypted-share image 2, (c) encrypted-share image 3, and (d) encrypted-share image 4

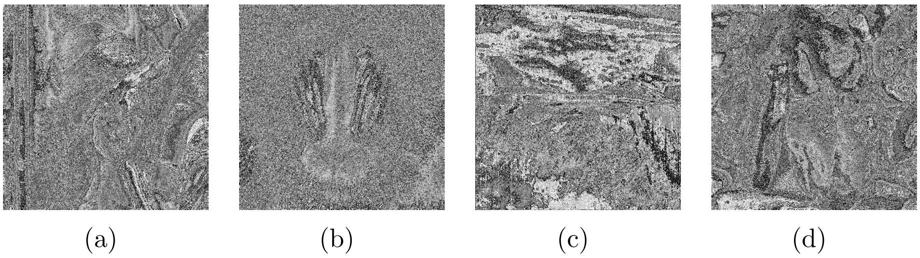


Figure 5. Experimental results of the encrypted-recover images : (a) Lena, (b) Baboon, (c) F-16, and (d) Peppers

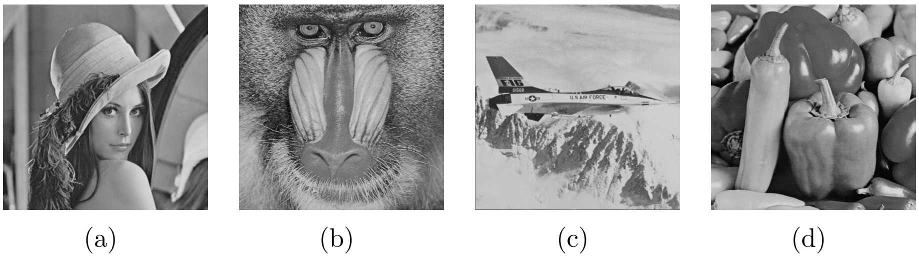


Figure 6. The recover images : (a) Lena, (b) Baboon, (c) F-16, and (d) Peppers

5. **Analysis of security and computational complexity.** In this section, we analyze the security and computational complexity of our proposed scheme. First, we show that our scheme can ensure data confidentiality and data integrity, and prevent statistical and entropy-based attacks. Then, the computational performance of our scheme is low.

5.1. **Security analysis.** .

A. Data confidentiality. Since our scheme uses RSA encryption to ensure data confidentiality, i.e., the encrypted-share images $EnSI^1, EnSI^2, \ldots, EnSI^n$ are encrypted by the user's public key, the security of our scheme is the same as RSA encryption. If an illegal user wants to decrypt the encrypted-share image, he/she first needs to solve the prime factorization problem, i.e., break the RSA assumption. Therefore, data confidentiality is guaranteed in our proposed scheme.

B. Data integrity. In the recovery phase, user U downloads at least t encrypted-share images from multiple cloud storage systems; he/she can retrieve the encrypted-recover image RI that is encrypted by the user's public key. Finally, the user can obtain the secret image I by employing the corresponding private key to decrypt RI. Once user U wants to check data integrity, he/she can download t encrypted-share images twice. Then user U retrieves the encrypted-recover images RI_1 and RI_2 by performing Step R1 of Subsection 3.3. Finally, user U checks whether RI_1 is the same as RI_2 or not to ensure data integrity. In this manner, although some encrypted-share images are broken or lost, the user still can check data integrity.

C. Statistical and entropy-based attacks. Here, we statistically analyze the security of our scheme, i.e., histogram distribution, correlation of the adjacent pixels, and information entropy.

C1. Histogram distribution. In this analysis, the gray-scale image "Lena"(512×512 pixels) was used in our scheme. The plain image is the secret image and the cipher images are encrypted-share images. In a secure encryption scheme, the histograms of the cipher images should be uniform. In Figure 7, we can see that the histogram of the cipher images seems fairly uniform and that these images are quite different from the plain image.

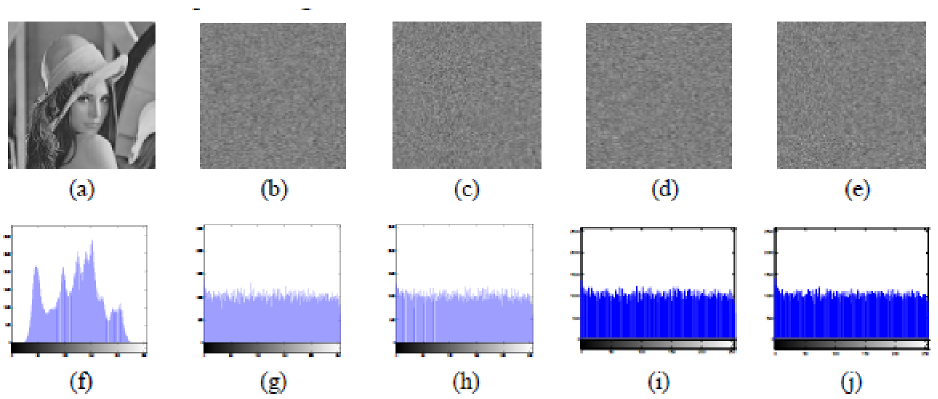


Figure 7. Histograms of the plain image and cipher images : (a) plain image, (b) cipher image 1, (c) cipher image 2, (d) cipher image 3, (e) cipher image 4, (f) histograms of plain image, (g) histograms of cipher image 1, (h) histograms of cipher image 2, (i) histograms of cipher image 3, and (j) histograms of cipher image 4

C2. Correlation of the adjacent pixels. Generally speaking, there are high correlations of the adjacent pixels in the plain image, while the adjacent pixels in the cipher images have low correlations. In order to measure the correlation performances in our scheme, correlation coefficient r is computed by the following formula:

$$r = \frac{\sum_{i=1}^{(H\times W)/2}(x_i - \frac{2}{(H\times W)}\sum_{i=1}^{(H\times W)/2} x_i) \times (y_i - \frac{2}{(H\times W)}\sum_{i=1}^{(H\times W)/2} y_i)}{\sqrt{\sum_{i=1}^{(H\times W)/2}(x_i - \frac{2}{(H\times W)}\sum_{i=1}^{(H\times W)/2} x_i)^2 \times \sum_{i=1}^{(H\times W)/2}(y_i - \frac{2}{(H\times W)}\sum_{i=1}^{(H\times W)/2} y_i)^2}}, \quad (3)$$

where x_i and y_i are adjacent pixels, and $H \times W$ is the size of the text image. Table 1 shows the distributions of the adjacent pixels in horizontal, vertical, and diagonal directions, respectively. We can see that the coefficient values r of the plain images are between 0.76 and 0.99, and the coefficient values r of the cipher images are all close to 0. This means the cipher images have low correlation, and our scheme is robust against statistical attacks.

C3. Information entropy. In 1948, Shannon introduced and defined the concept of "information entropy"[23]. Information entropy is a random variable that can express the chaotic degree of information content. In a robust encryption scheme, the value of information entropy should be higher. To measure the robustness of our proposed scheme, information entropy $H(I)$ was computed by the following formula:

$$H(T) =- \sum_{i=0}^{M} P(x_i)\log_2 P(x_i), \quad (4)$$

where T denotes the test image, x_i denotes the pixel value of text image T, and the $P(x_i)$ denotes the probability of x_i. Here, the test images are the plain and cipher images. Table 2 shows the results of information entropy in those images. For a gray-scale image in simulation, there are 256 symbols; therefore, the upper bound of information entropy is 8. In our scheme, the values of the information in the cipher images are all 7.99. This means our scheme is robust against entropy-based attack.

5.2. **Computational performance.** In this subsection, we compare the performance of a normal RSA scheme and our scheme. According to [24], we know that one secret

Table 1. The experimental results of correlation coefficients

Image		Lena	Baboon	F-16	Peppers
horizontal	Plain image	0.978910	0.915640	0.966250	0.987290
	Cipher image 1	0.000323	0.001678	0.000011	0.001072
	Cipher image 2	0.001933	0.000200	0.000909	0.000767
	Cipher image 3	0.000150	0.000556	0.002007	0.001782
	Cipher image 4	0.002473	0.001638	0.000688	0.005251
vertical	Plain image	0.987340	0.791370	0.964320	0.986130
	Cipher image 1	0.000241	0.000691	0.000661	0.001596
	Cipher image 2	0.000039	0.001293	0.000458	0.003469
	Cipher image 3	0.003137	0.000477	0.000280	0.000159
	Cipher image 4	0.000467	0.000436	0.000807	0.001518
diagonal	Plain image	0.965730	0.766800	0.937460	0.975350
	Cipher image 1	0.001379	0.003924	0.002771	0.003227
	Cipher image 2	0.001037	0.001065	0.000661	0.000659
	Cipher image 3	0.000907	0.001400	0.001602	0.000399
	Cipher image 4	0.002552	0.002093	0.001534	0.002408

Table 2. Results of information entropy in the test images

Entropy	Lena	Baboon	F-16	Peppers
Plain image	7.43	7.31	6.49	7.57
Cipher image 1	7.99	7.99	7.99	7.99
Cipher image 2	7.99	7.99	7.99	7.99
Cipher image 3	7.99	7.99	7.99	7.99
Cipher image 4	7.99	7.99	7.99	7.99

sharing scheme (SSS) is about 2,600 times faster than one asymmetric cryptosystem (RSA-1024). In a normal RSA scheme, the user needs to perform one asymmetric cryptosystem, i.e., one encryption operation and one decryption operation. In our scheme, we use the homomorphic property of the RSA cryptosystem and Shamirs secret sharing scheme to implement this scheme. Therefore, the user performs only one SSS and one decryption operation. As the above result shows, our proposed scheme can reduce the computation cost significantly compared to the normal RSA scheme.

6. **Conclusions.** In this paper, we have combined SSS and RSA technologies to accomplish a novel and effective encryption scheme for secret image stored in multiple cloud storage systems. Our proposed scheme can achieve the basic security requirements, such as data confidentiality, data integrity, and high robustness. In addition, this scheme allows users to check the integrity of secret image by her/himself, even broken or lost images stored in cloud storage systems. Analyses and simulations demonstrate that our scheme can be effective against statistical and entropy-based attacks, and the computational cost is lower. Therefore, our scheme is effective and practical for use in a cloud storage system.

REFERENCES

[1] Z. Xiao and Y. Xiao, Progressive watermarking techniques using genetic algorithms, *IEEE Communications Surveys and Tutorials*, vol. 15, no. 2, pp. 843-859, 2013.

[2] D.S.L. Wei, S. Murugesan, S.Y. Kuo, N. Kshirasagar, and D. Krizanc, Enhancing data integrity and privacy in the cloud: an agenda, *Computer*, vol. 46, no. 11, pp. 87-90, 2013.

[3] C.K Chu, W.T. Zhu, S.Y. Kuo, J. Han, J.K Liu, J. Xu, and J. Zhou, Security Concerns in Popular Cloud Storage Services, *IEEE Pervasive Computing*, vol. 12, no. 4, pp. 50-57, 2013.

[4] S.A Marc, Toward Efficient Data Access Privacy in the Cloud, *IEEE Communications Magazine*, vol. 51, no. 11, pp. 39-45, 2013.

[5] T. Mather, S. Kumaraswamy, and, S. Latif, *Cloud Security and Privacy: An Enterprise Perspective on Risks and Compliance*, O'Reilly Press, USA, 2009.

[6] B.R. Kandukuri, R.V. Paturi, and A. Rakshit, Cloud Security Issues, *Proc. of the IEEE International Conference on Services Computing*, pp. 21-25, 2009.

[7] T.Y. Wu, J.S. Pan, and C.F. Lin, Improving Accessing Efficiency of Cloud Storage Based on Deduplication and Feedback Scheme, *IEEE Systems Journal*, vol. 8, no. 1, pp. 208-218, 2014.

[8] C. Curino, E.P.C. Jones, R.A. Popa, N. Malviya, E. Wu, S. Madden, H. Balakrishnan, and N. Zeldovich, Relational Cloud: A Database-as-a-Service for the Cloud, *Proc. of the 5th Biennial Conference on Innovative Data Systems Research*, pp.235-240, 2011.

[9] L. Zhou, V. Varadharajan, and M. Hitchens, Achieving Secure Role-Based Access Control on Encrypted Data in Cloud Storage, *IEEE Trans. on Information Forensics and Security*, vol. 8, no. 12, pp.1947-1960, 2013.

[10] U. Somani, K. Lakhani, and M. Mundra, Implementing Digital Signature with RSA Encryption Algorithm to Enhance the Data Security of Cloud in Cloud Computing, *Proc. of the 1th International Conference on Parallel, Distributed and Grid Computing*, pp.211-216, 2010.

[11] J. Hur, Improving Security and Efficiency in Attribute-Based Data Sharing, *IEEE Trans. on Knowledge and Data Engineering*, vol. 25, no. 10, pp.2271-2282, 2013.

[12] Y.W. Kao, K.Y. Huang, H.Z. Gu, and S.M. Yuan, uCloud: a User-Centric Key Management Scheme for Cloud Data Protection, *IET Information Security*, vol. 7, no. 2, pp. 144-154, 2013.

[13] C. Wang, B. Zhang, K. Ren, and J.M. Roveda, Privacy-Assured Outsourcing of Image Reconstruction Service in Cloud, *IEEE Trans. on Emerging Topics in Computing*, vol. 1, no. 1, pp. 166-177, 2009.

[14] K. Yang and X. Jia, An Efficient and Secure Dynamic Auditing Protocol for Data Storage in Cloud Computing, *IEEE Trans. on Parallel and Distributed Systems*, vol. 24, no. 9, pp.1717-1726, 2013.

[15] C. Wang, Q. Wang, K. Ren, and W. Lou, Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing, *Proc. of IEEE Infocom*, pp. 1-9, 2010.

[16] C. Wang, S.S.M. Chow, Q. Wang, K. Ren, and W. Lou, Privacy-Preserving Public Auditing for Secure Cloud Storage, *IEEE Trans. on Computers*, vol. 62, no. 2, pp. 362-375, 2013.

[17] C. Wang, K. Ren, W. Lou, and J. Li, Toward Publicly Auditable Secure Cloud Data Storage Services, *IEEE Network*, vol. 24, no. 4, pp. 19-24, 2010.

[18] A. Shamir, How to Share a Secret, *Communications of the ACM*, vol. 22, no. 11, pp. 612-613, 1979.

[19] R.L. Rivest, A. Shamir, and L. Adleman, A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, *Communications of the ACM*, vol. 21, no. 2, pp. 120-126, 1978.

[20] J. Gordon, Strong RSA key, *IET Electronics Letters*, vol. 20, no. 12, pp. 514-516, 1984.

[21] J.H. Moore, Protocol Failures in Cryptosystems, *Proceedings of the IEEE*, vol. 76, no. 5, pp. 594-602, 1988.

[22] H.C.A. Tilborg and S. Jajodia (eds.), *Encyclopedia of Cryptography and Security*, Springer, USA, 2011.

[23] C.E. Shannon, A Mathematical Theory of Communication, *The Bell System Technical Journal*, vol. 27, no. 3, pp.379-423, 1948.

[24] B. Schneier, *Applied cryptography, protocols, algorithms, and source code in C*, John Wiley and Sons Inc., New York, U.S.A., 1996.