A Secure and Efficient Scheme for Cloud Storage against Eavesdropper

Jian Liu, Huimei Wang, Ming Xian, and Kun Huang

State Key Laboratory of Complex Electromagnetic Environment Effects on Electronics and Information System, National University of Defense Technology, Changsha, 410073, China
ljabc730@nudt.edu.cn

Abstract. Cloud storage system, which can be viewed as a large collection of individually unreliable storage nodes, is potential to be faced with the threat of data loss and leakage, due to node failure and eavesdropped by an intruder. As a solution, secret sharing scheme stores the data redundantly across the distributed storage system(DSS) and it is able to protect data security against ℓ-eavesdropper without need of secret key management mechanism, however, it do not provide regeneration property. Combining the regenerating code with the secret sharing scheme is an effective approach to address this drawback, yet all the schemes that have been proposed in previous work are conducted under the perfect-security criterion and leads to an unaffordable loss of the storage capacity while the number of observed nodes ℓ get close to threshold k. In this paper we adopt the weak-security criterion and give a formal description of "Secure DSS against an ℓ-eavesdropper". Applying a secure hash function and concatenated with the Product-Matrix minimum bandwidth regenerating(PM-MBR) code, our scheme significantly improves the secrecy capacity and keeps the loss of data rate constantly in a low level with any ℓ. As the analysis result indicates, our scheme, which provides sufficient security, repair efficiency and storage efficiency, is more suitable for practical systems. Moreover, we introduce another approach as an extension, which combines the All-Or-Nothing Transform with PM-MBR, and finally achieves a secure storage against ℓ-eavesdropper without loss of data rate.

Keywords: Cloud Storage, Eavesdropper, PM-MBR, Data Confidentiality, Hash Function, All-Or-Nothing Transform, Secrecy Capacity.

1 Introduction

Cloud storage system is now increasing attracting individuals and organizations to outsource their data from local to remote cloud servers. However, many consumers are still feel hesitant, since they lose their control on the data which maybe lost and leaked by incidents. To deal with node failure and compromised by an attacker, which leads to data loss and leakage, the system is desired to provide both storage reliability and confidentiality. In general, a cloud storage system is considered as a large-scale distributed storage system(DSS) which consists of many independent storage nodes.

For reliable storage, DSS stores the data redundantly(e.g., by using (n, k)RS code) on a collection of individually unreliable storage nodes, such that the data can be recovered from active nodes even if a small set of nodes fails. However, the repair-bandwidth is so high up to the size of original file when only one node fails in this scenario. A repair-efficient scheme, called *regenerating codes(RC)*, was proposed in [1], where the tradeoff between the storage capacity of single node and the repair-bandwidth was studied, furthermore codes that achieve two extreme points, which are known as *minimum storage regenerating* (MSR) codes and *minimum bandwidth regenerating* (MBR) codes, were introduced. Moreover, explicit construct methods for MBR and/or MSR codes that allow *exact repair*[24] were presented in [2–4].

Besides reliability, storage confidentiality is also a significant property for DSS. Paulo *et al.*[5] considered a scenario in which a large, private file is to be stored securely and there exists an *intruder* may gain access to some storage nodes in the DSS, but not all. Assuming that the *intruder* only can eavesdrop on compromised nodes but cannot modify the data stored on them, it's desirable to ensure that the *intruder* is unable to recover the whole file or any of its parts. A straightforward solution is that, first encrypt the file using a secret key and then partition the resulting cryptogram into multiple shares that can be spread over the storage nodes. However, such cryptographic solution introduces the need for secret key management mechanism[6], which increases the overall complexity and the resources demanded by the system. The secret sharing scheme provides an effective solution without secret key management. In a secret sharing scheme, one divides a secret into shares, and a threshold number of shares is sufficient to recover the original secret but any number of shares(obtained by the intruder) smaller than the threshold reveal no information about the secret[5, 7, 8, 16]. Some other practical application of the combination of secret sharing scheme and erasure codes was proposed in [9, 10], these schemes applied cryptography but without need of key management and distribution. Even though all the schemes above provide reconstruction and security of data shares, they do not provide the property of regenerating the share as was stored in the failed node.

To address this drawback, combining *RC* with the secret sharing scheme is an effective way. It is noted that schemes following this way have been also studied in [11–14]. S.Pawar *et al.*[11] gave the upper bound of secrecy capacity for secure (n, k, d)-DSS against passive eavesdroppers and proposed achievable approach, based on nested MDS and RSKR-repetition codes, in the bandwidth-limited regime for repair degree $d = n - 1$. Further more, literature [13, 14] proposed *information-theoretically secure* MBR and/or MSR codes that achieve the secrecy capacity in [11]. Moreover, Rawat *et al.*[12] gave tighter bound on the secrecy capacity of a DSS at the MSR point, and presented an approach based on Gubidulin precoding to achieve the upper bound for certain system parameters. However, all these schemes achieve the *perfect-security criterion*, that is, intruder eavesdropping $\ell < k$ nodes gets *no information* of the original file stored.

By mixing a certain number of random symbols with original data before encoded by RC, these schemes lead to a decrease of the file size that can be securely stored in the DSS and the loss of secrecy capacity is unaffordable when ℓ get close to k. Take [13] for example, a $(n = 6, k = 3, d = 4)$ PM-MBR is applied, the storage capacity is 9β, in the presence of an intruder that can eavesdrop the data on two nodes, we need the ratio of random symbols up to 7/9 of the storage capacity to be added to achieve the *perfect-security*, and thus the data rate 2/9 is unbearable. Considering the fact that *perfect security criterion* is too strict and leads to above disadvantages, we adopt the *weak-security criterion*[15] in our work instead. In case of *weak-security criterion*, intruder cannot get any "*meaningful*" information of the stored data symbols when eavesdropping $\ell(< k)$ storage nodes.

In this paper, we focus on designing an efficient scheme for DSS, which provides not only data reliability but also data security against passive eavesdroppers. Taking the PM-MBR code[3] as the basis of our scheme, it satisfies the *regeneration property*(i.e., can regenerate a lost share with low bandwidth). Using a hash function or AONT[22] as the preprocessing procedure before data symbols are encoded by minimum bandwidth regenerating code, our scheme achieves the *weak security criterion*, which is sufficient in practical distributed storage scenario. Simultaneously the data capacity securely stored in the DSS is significantly improved, and the loss of data rate is kept constantly in a low level(equal to zero in case of AONT) for any $\ell(< k)$ with fixed (n, k, d).

Similar problem has been considered in the research field of multicast-network applying network coding[19]. [15, 22, 23] showed that weakly secure network coding can achieve the security with a higher multicast rate compared with perfect secure network coding[20, 21]. Inspired by the achievement of [15, 22, 23], their ideology is introduced into our research for building a secure distributed storage system against eavesdropper with high secrecy capacity. However, we use totally different models and methods in our work. To our best knowledge, few papers strived in this direction applying weak security criterion to DSS.

Contributions: Our main contribution in this paper is to provide a secure and efficient scheme, which exploits the Product-Matrix framework[3], for DSS. The proposed scheme provides the following guarantees:

- *Weak-Security Property:* We assure that an intruder who can eavesdrop any $\ell(< k)$ out of n storage nodes is unable to recover any individual original data symbol. To be viewed as a secret sharing scheme, however, our scheme does not apply secret key management and distribution mechanism.
- *Reconstruction Efficiency:* Since the regenerating code utilized by us satisfies MDS property, our proposed scheme is resilient to node failures, that is, a data collector is able to reconstruct the original data file as long as there are k out of n storage nodes active.
- *Regeneration Efficiency:* We combine minimum bandwidth regenerating code with secret sharing scheme, thus ensure that our scheme not only provides security guarantees, but also satisfies regeneration efficiency, i.e., low repair-bandwidth when node failure happens.

– *Secure Storage Efficiency(High Secure Data Rate):* We apply a secure hash function in the preprocessing procedure, and then concatenated with PM-MBR code. This approach is effective to improve secure data capacity, and keep the loss of data rate constantly in a low level. Moreover, another construction method with All-Or-Nothing Transform(AONT) is presented, which can achieve the secure storage without loss of data rate.

Organization: The rest of this paper is organized as follows. In Section 2 we introduce the system model, intruder model and some preliminaries for our scheme. Our proposed scheme is detailed described in Section 3 and evaluated in Section 4. In Section 5 we briefly present another construction method and analyze the essence of its security. Finally, we conclude this paper Section 6.

2 Model and Preliminaries

2.1 System Model

We consider the cloud storage to be a large-scale DSS, which consists of three components, i.e., the source node s, n active storage nodes and data collector DC. There's an incompressible data file $\mathcal{F}$ of $\mathcal{M}$ symbols(each belonging to a finite field $\mathbb{F}_q$) in the source node s. We assume that each storage node has a storage capacity of α symbols and is individually unreliable and may fail over time.

The source node s split, encode the file $\mathcal{F}$ and then distribute the coded result to the n connected storage nodes $\{v_1, v_2, ...v_n\}$. Any DC who can connect to any k out of n active nodes should be able to retrieve the $\mathcal{M}$ symbols and reconstruct the original file $\mathcal{F}$. We term this the *MDS property* of the DSS. To maintain the k-out-of-n MDS property, failed node must be immediately replaced by newcomer with same storage capacity α. In our work, we focus on the case of symmetrical repair, where the newcomer connects to arbitrary d active nodes out of the remaining active nodes and downloads equal amount of symbols, say β, from each. The repair degree d is a system parameter satisfying $k \leq d \leq n-1$. The corresponding repair bandwidth of the system is defined as $\gamma = d\beta$ in this paper. Thus we define such a DSS as $\mathcal{D}(n, k, d)$. For instance, the DSS depicted in Fig.1 corresponds to $\mathcal{D}(4, 2, 3)$ which is operating at $(\alpha, \gamma) = (2, 3)$ with *functional repair*, the failed node v_1 is replaced by the newcomer v_5 in this scenario.

Moreover, for the reason that the *functional repair*[24] has some inherent security shortcomings for DSS in the presence of an eavesdropper, in this paper our presented scheme employs the *exact repair* where the newcomer regenerates an exact copy of the lost data and thus $\alpha = d\beta$. Specifically, the *PM-MBR code* proposed by Rashmi *et al.*[3] is applied. Besides, we denote the storage capacity of the DSS as $\mathcal{L}$, which was derived by [1] and:

$$\mathcal{L} = \sum_{i=1}^{k} \min\{(d-i+1)\beta, \alpha\} \tag{1}$$

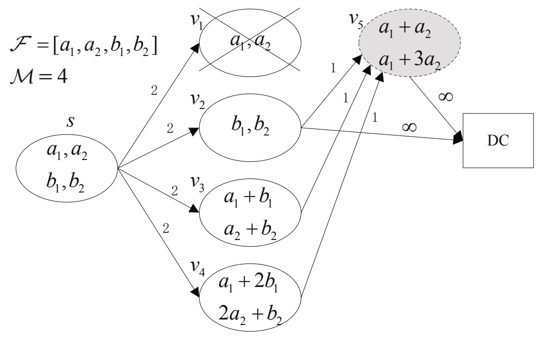

Fig. 1. $\mathcal{D}(4,2,3)$ under repair

Note that the right-hand side of Eq.(1) can be reduced to $\sum_{i=0}^{k}[(d-i+1)\beta]$ in case where PM-MBR code is applied[3].

2.2 Intruder Model

We consider an (ℓ_1, ℓ_2) eavesdropper, which can access the stored data of nodes in set ε_1, and additionally can access both the stored and downloaded data of the nodes in set ε_2, with $|\varepsilon_1| = \ell_1$ and $|\varepsilon_2| = \ell_2$. The eavesdropper is passive and can only read the data on the observed nodes without modifying it. This is the same eavesdropper model considered in [12]. For the MBR code used in this paper, we have $d\beta = \alpha$, i.e., a replacement node stores all the data that it downloads during its repair. Thus an eavesdropper does not obtain any extra information from the data that is downloaded for repair. Without loss of generality, we can assume that $\ell_1 = \ell(\ell < k), \ell_2 = 0$, and then the intruder model is simplify to the ℓ-eavesdropper. In addition, the eavesdropper is assumed to have complete knowledge of the storage and coding schemes employed by the DSS. As a result, the intruder can choose any ℓ nodes from the initial storage nodes and/or the replacement nodes. For example in Fig.1 with $\ell = 1$, the replacement node v_5 is compromised by the intruder and shown in grey background.

2.3 Security Criterion

Here we first introduce *perfect security criterion* and then present our definition of the secure DSS against the ℓ-eavesdropper in this subsection.

Let $S = [s_1, s_2, s_3, ..., s_L]^T$ be a random vector uniformly distributed over $\mathbb{F}_q^L$, representing the incompressible data file at the source node. Each symbol of S denoted by $s_i(i = 1, 2, ..., L)$ is independent random variable uniformly distributed over $\mathbb{F}_q$. Let $H(S)$ denotes the entropy of the random variable S. The S is encoded into n shares $c_i \in \mathbb{F}_q^\alpha$. For each $i \in \{1, 2, .., n\}$, a share c_i is stored in node v_i. Let B be a collection of k active nodes randomly chosen from all n storage nodes, and define $C_B := \{c_i : v_i \in B\}$. Similarly, we let E be the collection of nodes that can be observed by the eavesdropper, and $|E| = \ell < k$ in our work, thus define $C_E := \{c_i : v_i \in E\}$.

Then the reconstruction property (or k-out-of-n MDS property) of the DSS, can be written as:

$$H(S|C_B) = 0, \qquad |B| = k. \tag{2}$$

and the perfect security condition implies:

$$H(S|C_E) = H(S), \qquad |E| = \ell < k. \tag{3}$$

The Eq.(3) can be interpreted that the eavesdropper cannot get any information about the original vector $S = [s_1, s_2, s_3, ..., s_L]^T$, even she is able to obtain the data shares on ℓ compromised nodes. As mentioned above, this security criterion is too strict and unnecessary in practical scenario. Notice the fact that an intruder who get the $s_i \oplus s_j (i \neq j)$ is also unable to recover the symbols s_i and s_j, i.e., $I(s_i; s_i \oplus s_j) = I(s_j; s_i \oplus s_j) = 0$, because these symbols are uniformly, independently and randomly distributed over $\mathbb{F}_q$. Therefore, it is sufficient that an intruder cannot get any "*meaningful*" information of each symbol in original vector S to guarantee the data secure against the ℓ-eavesdropper($\ell < k$), which is called the *weak-security* criterion[15]. We present the definition of a secure DSS following such criterion:

Definition 1. *(Secure DSS against an ℓ-eavesdropper): A DSS is said to be secure against an ℓ-eavesdropper, if, for any set E of size $\ell < k$,*

$$I(s_i; C_E) = 0, (i = 1, 2, ..., L_s) \tag{4}$$

Where s_i denotes the symbol of the data file S, and C_E represents data observed by the eavesdropper. Besides, we denote the file size that can be secure stored in the DSS as L_s.

Under the *weak-security* condition[15], our proposed scheme can achieve a higher secure data capacity as discussed in Section 4.

2.4 Product-Matrix MBR Code

Rashmi *et al.*[3] proposed the first construction of general and optimal exact-regenerating code such as (a)(n, k, d) MBR code for all values of (n, k, d),and (b)(n, k, d) MSR code for all values of (n, k, d) where $d \geq 2k - 2$. In general, in order to obtain a lower repair bandwidth, all n, k, d should satisfy $k \leq d \leq n-1$. As the basis of our work, we briefly review the Product-Matrix MBR code here.

As assumed in [13], we set $\beta = 1$. It is reasonable since that any higher value of β can be obtained by a simple concatenation of the $\beta = 1$ code. Thus the PM-MBR code with $\beta = 1$ and $\alpha = d$ can be interpreted as an $(n \times \alpha)$ code matrix C, where each row corresponds to one storage node of the DSS, i.e., the α elements in the i^{th} row represent the α symbols stored in node $v_i (i = 1, 2, ..., n)$. The code matrix C is a product of two matrices: a fixed encoding matrix $\Psi_{n\times d}$ and a message matrix $M_{d\times\alpha}$, i.e., $C_{n\times\alpha} = \Psi_{n\times d} \cdot M_{d\times\alpha}$.

The encoding matrix $\Psi_{n\times d}$ has the form as $\Psi_{n\times d} = [\Phi_{n\times k} \quad \Delta_{n\times(d-k)}]$ where the matrices $\Phi_{n\times k}$ and $\Delta_{n\times(d-k)}$ are chosen to satisfy (i)any k rows of Φ are linearly independent, and (ii)any d rows of Ψ are linearly independent, thus, a Vandermonde or Cauchy Matrix is applicable.

The message matrix $M_{d\times\alpha}$ contains the symbols of the data file to be stored in a redundant fashion. From Eq.(1) we get that the stored symbols amount to $\mathcal{L} = \frac{k(2d-k+1)}{2} = \frac{k(k+1)}{2} + k(d-k)$ in the DSS employing the PM-MBR code. The message matrix $M_{d\times\alpha}$ is of the following form

$$M_{d\times\alpha} = \begin{bmatrix} U_{k\times k} & V_{k\times(d-k)} \\ \\ V^t_{k\times(d-k)} & O_{(d-k)\times(d-k)} \end{bmatrix}$$

The $U_{k\times k}$ in the above expression denotes a symmetric matrix, thus the $\frac{k(k+1)}{2}$ components in the upper-triangular half of the matrix are filled up by $\frac{k(k+1)}{2}$ distinct message symbols drawn from the $\mathcal{L}$ message symbols. The remaining $k(d-k)$ message symbols are used to fill up the matrix $V_{k\times(d-k)}$. The V^t denotes the transpose of matrix $V_{k\times(d-k)}$, and the $O_{(d-k)\times(d-k)}$ denotes the zero matrix with all zero components. To keep things simple, we use these denotations without the subscript in the rest of this paper.

3 Our Proposed Scheme

3.1 Notation and Definition

The original data file of size L is represented by a vector $S = [s_1, s_2, ..., s_L]^T$, with $s_i \in \mathbb{F}_q, i = 1, 2, ..., L$. Supposing that the file has been optimally compressed, then the symbols $s_i (i = 1, 2, ..., L)$ are independent random variables uniformly distributed over the finite field $\mathbb{F}_q$. Let K denotes a set of random symbols and $|K| = L_r$, and each symbol is chosen independently and uniformly across the elements of $\mathbb{F}_q$. In addition, we make use of suitable one-way function, i.e., a secure hash function defined as follow, in our scheme.

Definition 2. *(Secure Hash Function): The function $h(key, \bullet) : key \times X \to Y$ where $key \in \mathbb{F}_q$, for $\forall X \in \mathbb{F}_q^r (r \in \mathbb{N}^*)$, the result $Y \in \mathbb{F}_q$, is secure if the following conditions satisfy:*

(1)Given a hash value $y \in \mathbb{F}_q$, it's hard to find any $x \in \mathbb{F}_q^r (r \in \mathbb{N}^*)$ in polynomial time such that $y = h(key, x)$;

(2)Given any $x \in \mathbb{F}_q^r (r \in \mathbb{N}^*)$,the hash value $y = h(key, x)$ can be easily got in polynomial time;

(3)Given any $x \in \mathbb{F}_q^r (r \in \mathbb{N}^*)$, it's hard to find $x' \in \mathbb{F}_q^r (r \in \mathbb{N}^*)$ that have the same hash value in polynomial time, i.e., $h(key, x') = h(key, x)$.

To keep things simple, we define that $h(x) = h(x, x)$ with $x \in \mathbb{F}_q$ in the following subsection.

Besides, we denote the i^{th} row of the encoding matrix Ψ as ψ_i. In order to meet the requirement of Ψ, we adopt the Vandermonde matrix to be the encoding matrix. The message matrix is represented by M. All the elements of matrix Ψ belong to the finite field $\mathbb{F}_q$, thus, $q \geq n$. Further, we set $\beta = 1$ in the scenario of $\mathcal{D}(n, k, d)$ in which our scheme executes.

3.2 Detailed Scheme

Our scheme is consisted of four procedures shown as follows:

1) Preprocessing Procedure:

step 1: Generate a set K that contains $L_r = d - k + 1$ random symbols $\{k_1, k_2, ...k_{L_r}\}$, where each symbol is independently, randomly and uniformly distributed over $\mathbb{F}_q$;

step 2: Let S' be a set with cardinality $L_s(< L)$, and the elements of S' are the symbols drawn from the incompressible data file $S = [s_1, s_2, ..., s_L]^T$, here we set $S' = \{s_1, s_2, ...s_{L_s}\}$ without loss of generality. In the scenario of $\mathcal{D}(n, k, d)$, $L_s = \mathcal{L} - L_r = \frac{k(2d-k+1)}{2} - (d - k + 1)$;

step 3: Let $sk = h(k_1, k_2||k_3||...||k_{L_r})$, where $k_2||k_3||...||k_{L_r}$ denotes the concatenation of the random symbols $k_i(i = 2, 3, ...L_r)$. Utilizing the hash value sk to preprocess the original data symbols, we get the result $P = \{p_1, p_2, ...p_{L_s}\}$ where $p_1 = s_1 \oplus h(sk)$, $p_i = s_i \oplus h(sk, s_1||s_2||...||s_{i-1})(i > 1, i \in \mathbb{N}^*)$;

2) Encoding and Distributing Procedure:

step 1: Note that $L_s + L_r = \mathcal{L}$, we populate the message matrix M with elements in P and K. To be specific, place the symbols of P into the first $k - 1$ rows and hence first $k - 1$ columns of the symmetric matrix M, the rest position of submatrix U and V of M is filled with L_r random symbols of K(an example is shown in Fig.2);

step 2: Choose n elements from $\mathbb{F}_q$ and then generate a Vandermonde matrix to be the encoding matrix Ψ. According to the encoding method of PM-MBR, we multiply the matrix Ψ and M to get the Product-Matrix $C = \Psi \cdot M$.

step 3: Extracting each row(denoted by c_i) of the Product-Matrix C as a share(of size d) of encoded data, the source node distributes them to the corresponding storage node v_i, $i = 1, 2, ...n$.

3) Regeneration Procedure:

Supposing that node $v_f(f \in [n])$ fails at a time, the regeneration process of our scheme is the same as that in traditional PM-MBR code[3]:

step 1: Notice that the share stored on the node v_f is $c_f = \psi_f M$. The replacement for the failed node f connects to an arbitrary set $\{h_i | 1 \leq i \leq d\}$ of d remaining nodes.

step 2: Each of these d nodes passes on the inner product $(\psi_{h_i} M)\psi_f^t$ to the replacement node. Thus from these d nodes, the replacement node obtains the $d = \alpha$ symbols, i.e., $C_{rev} = \Psi_{rev} M \psi_f^t$, where $\Psi_{rev} = [\psi_{h_1}, \psi_{h_2}, ..., \psi_{h_d}]$ is invertible by construction.

step 3: Replacement node performs matrix inversion on Ψ_{rev} and multiplies Ψ_{rev}^{-1} with C_{rev}. Thus $M\psi_f^t$ can be recovered, since M is symmetric, $(M\psi_f^t)^t = \psi_f M$ is precisely the data stored in the node prior to failure.

4) Reconstruction Procedure:

When a data collector try to reconstruct the original data file stored in the (n, k, d)-DSS, it firstly perform the same action as traditional PM-MBR code[3], then recover original symbols:

step 1: Data collector connects to k out of n active storage node, let $\Psi_{DC} = [\Phi_{DC} \quad \Delta_{DC}]$ be the corresponding k rows of Ψ. Thus the data collector gains the symbols $\Psi_{DC}M = [\Phi_{DC}U + \Delta_{DC}V^t \quad \Phi_{DC}V]$. Being a submatrix of Vandermonde matrix, Φ_{DC} is nonsingular. Hence, by multiplying the matrix $\Psi_{DC}M$ on the left by Φ_{DC}^{-1}, one can recover first the matrix V and subsequently the matrix U. Thus, all symbols in set K and P are gained by data collector;

step 2: Compute the $sk = h(k_1, k_2||k_3||...||k_{L_r})$ from K, and then the original symbols in set $S' = \{s_1, s_2, ...s_{L_s}\}$ can be got in the way: $s_1 = p_1 \oplus h(sk)$, $s_i = p_i \oplus h(sk, s_1||s_2||...||s_{i-1})$, $(2 \le i \le L_s$ and $i \in \mathbb{N}^*)$. Finally, the original data file stored in the system is reconstructed.

3.3 An Example

Detailed description of our scheme above should be tedious and stuffy, however, in order to make our scheme seems more concrete, we illustrate it with an example similar to [13] in this subsection.

Example 1. Let $(n, k, d) = (6, 3, 4)$, then with $\beta = 1$,we get $\alpha = d = 4$, $L_r = 2$, $L_s = 7$ and $\mathcal{L} = 9$. Our scheme is designed over the finite field $\mathbb{F}_7$. The (6×4) encoding matrix Ψ is chosen as a Vandermonde matrix with its i^{th} row as $\psi_i = [1 \quad i \quad i^2 \quad i^3]$.

As depicted in Fig.2, the original data symbols $s_1, s_2, ..., s_7$ drawn from the incompressible data file F are first preprocessed with the random symbols in K, and then placed into the first two rows and first two columns of the message matrix $M_{4\times4}$. In the end, the matrices U and V are populated by the preprocessed symbols in $P = \{p_i\}_{i=1}^{7}$ and random symbols in $K = \{k_1, k_2\}$:

$$U = \begin{bmatrix} p_1 & p_2 & p_3 \\ p_2 & p_4 & p_5 \\ p_3 & p_5 & k_1 \end{bmatrix}, \qquad V = \begin{bmatrix} p_6 \\ p_7 \\ k_2 \end{bmatrix}$$

Further, $M_{4\times4}$ is multiplied on the left with the Vandermonde matrix $\Psi_{6\times4}$ and thus product-matrix $C_{6\times4}$ can be generated. Each row of $C_{6\times4}$ denoted by $\psi_i M$ corresponds to the share c_i stored on storage node $v_i (i = 1, 2, ..., 6)$.

4 Discussion

In this section, we evaluate the security property of $\mathcal{D}(n, k, d)$ applying our scheme, as well as its advantages in the aspects of secure data capacity and repair bandwidth over previous work. Then the computation cost of the proposed solution is compared against original PM-MBR scheme.

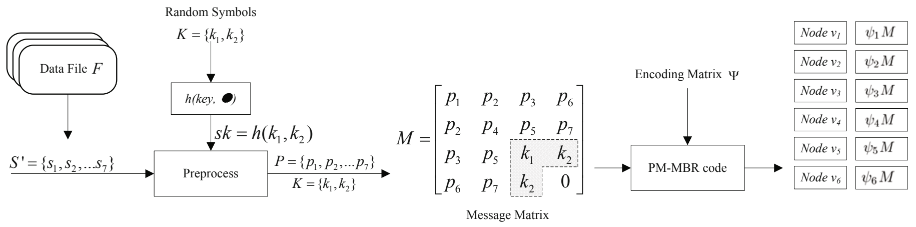

Fig. 2. An example with $(n, k, d) = (6, 3, 4)$

4.1 Security Analysis

To prove that our presented scheme is secure against an ℓ-eavesdropper($\ell < k$) in the (n, k, d)-DSS, we make use of the method appeared in [13] at first.

Lemma 1. *For any C_E representing the shares $\{c_i : v_i \in E\}$ stored on the node collection $E(|E| = \ell < k)$ eavesdropped by the intruder, no information about the random symbols set K is revealed, i.e., the mutual information between the random symbols and the eavesdropped shares is zero, that is*

$$I(K; C_E) = 0 \tag{5}$$

Proof. To prove Eq.(5), our scheme can be interpreted as a particular case of the construction method in [13], where $\ell = k - 1$, $\mathcal{E} = C_E$, and $\mathcal{U} = K$. Since original data symbol $s_i (i = 1, 2, ..., L_s)$ in S is independently, randomly and uniformly distributed over $\mathbb{F}_q$, and so is the hash value, thus $p_1 = s_1 \oplus h(sk)$ and $p_i = s_i \oplus h(sk, s_1||s_2||...||s_{i-1})(2 \le i \le L_s, i \in \mathbb{N}^*)$ all satisfy independent, random and uniform distribution over $\mathbb{F}_q$ as well. Then the preprocessed symbols set $P = \{p_i\}_{i=1}^{L_s}$ can be equivalent to the random symbols set $R = \{r_i\}_{i=1}^{L_s}$ denoted in previous literature[13].

Based on the above analysis, our proof proceeds in the same manner as [13](Section II), that is, $H(P|C_E, K) = 0$, $H(C_E) \le L_s$ proved firstly, and finally $I(K; C_E) = 0$ can be obtained. We do not present the details here due to lack of space.

Thus the ℓ-eavesdropper cannot obtain any information about the set K. □

Theorem 1. *The ℓ-eavesdropper($\ell < k$) cannot get any "meaningful" information of the original data symbols from C_E, i.e., the mutual information between each $s_i(i = 1, 2, ...L_s)$ and eavesdropped shares set C_E, that is*

$$I(s_i; C_E) = 0, (i = 1, 2, ..., L_s) \tag{6}$$

Proof. As shown in *Lemma 1*, the ℓ-eavesdropper cannot obtain any information of the K, the mutual information between $sk = h(k_1, k_2||...||k_{L_r})$ and C_E comes to zero. This is determined by the property(3) of the function $h(key, \bullet)$ defined in *Definition 2*. Similarly, we can get $I(h(sk); C_E) = 0$;

Next, we will show that the mutual information between original symbol $s_i(i = 1, 2, ...L_s)$ and C_E is zero. We use the inductive method here, and take

$i = 1$ in the beginning, $s_1 = p_1 \oplus h(sk)$, since p_1 is an independent and random variable uniformly distributed in $\mathbb{F}_q$,

$$I(s_1; C_E) = I(p_1 \oplus h(sk); C_E) = 0 \tag{7}$$

can be obviously obtained. For $i \geq 2$, assuming that $I(s_{i-1}; C_E) = 0$, since s_i and s_j are independent for $i \neq j$, and that the hash function is secure against collision(property(3) in *Definition 2*), thus we see that $I(h(sk, s_1||s_2||...||s_{i-1}); C_E) = 0$. Finally we get that

$$I(s_i; C_E) = I(p_i \oplus h(sk, s_1||s_2||...||s_{i-1}); C_E) = 0, \quad (2 \leq i \leq L_s, i \in (N)^*) \tag{8}$$

since p_i satisfies independent, random and uniformly distribution over $\mathbb{F}_q$.

Thus, equation(6) can be achieved for any $i \in [L_s]$, i.e., and the $\mathcal{D}(n, k, d)$ applying our scheme is secure against ℓ-eavesdropper as shown in *Definition 1.* □

In essence, notice that the input parameter of the secure hash function $p_i = h(key, \bullet)$ differs from each other for any $i = 1, 2, ...L_s$, and each element of $P = \{p_1, p_2, ...p_{L_s}\}$ is "*encrypted*" with different key(similar to *one-time pad*). Thus an intruder is unable to obtain any information of each original data symbol s_i without knowledge about the hash value sk(see *Lemma 1*), even though she has obtained some preprocessed symbols $p_i(i \in [L_s])$.

4.2 Repair-Bandwidth and Secrecy Capacity Analysis

Obviously, our scheme satisfies the MDS property(see Section 1), thus we analyze its repair and storage performance in such scenario: an $\mathcal{D}(n, k, d)$ with each storage node capacity up to α, the *newcomer* connects to d nodes out of the active ones when node failure happens, thus repair-bandwidth $\gamma = d\beta$. In such scenario, user accessing any k out of n storage nodes can recover the original data file, however, access to any $\ell < k$ does not leak any information about each original data symbol. So our proposed method can be viewed as a secret sharing scheme.

Before the analysis, we define data rate as follows at first:

$$\mathcal{R} = \frac{\mathcal{L}_{sec}}{\mathcal{L}} = \frac{\mathcal{L}_{sec}}{\sum_{i=0}^{k} \min\{(d - i + 1)\beta, \alpha\}} \tag{9}$$

where $\mathcal{L}_{sec}$ denotes the secrecy capacity of the system, and $\mathcal{L}$ denotes the storage capacity of $\mathcal{D}(n, k, d)$ using symmetrical repair.

Firstly, we will show that our scheme is repair efficiently compared to previous secret sharing schemes. Paulo F. Oliveira *et al.*[5] present a coding method and realize multi-secret sharing scheme, their schemes improved the secrecy capacity compared to Shamir's (k, n)-threshold scheme[16] and Bessani's (k, L, n)-threshold scheme[8]. As shown in Fig.3, Paulo's scheme provides an secrecy capacity up to the storage capacity. However, all their repair-bandwidth is so high up to the same as the Reed-Solomon coding scheme, that $\gamma = d\alpha = k\alpha$. Combined with MBR code, our secret sharing scheme provides the repair-bandwidth the same as minimum bandwidth regenerating code, that is, $\gamma = d\beta = \alpha$, which is much lower than schemes in [5, 8, 16]. Even though both the secrecy capacity and storage capacity of our scheme is slightly lower than Paulo's scheme*et al.*[5], as well as data rate is lower than 1, that is affordable and allowable in practise.

Various Schemes	Repair Bandwidth γ	Secrecy Capacity $\mathcal{L}_{sec}$	Storage Capacity $\mathcal{L}$	Secure Data Rate $\mathcal{R}$	Security Criterion
RS code	$k\alpha$	0	$k\alpha$	0	No
MBR code	α	0	$\left[kd-\binom{k}{2}\right]\beta$	0	No
Paulo's Scheme [5]	$k\alpha$	$k\alpha$	$k\alpha$	1	Weak
Shah's Scheme [13]	α	$\left[kd-\binom{k}{2}\right]\beta-\left[ld-\binom{\ell}{2}\right]\beta$	$\left[kd-\binom{k}{2}\right]\beta$	$1-\frac{\ell^2-(2d+1)\ell}{k^2-(2d+1)k}$	Perfect
Ours	α	$\left[kd-\binom{k}{2}\right]\beta-(d-k+1)\beta$	$\left[kd-\binom{k}{2}\right]\beta$	$1-\frac{2(d-k+1)}{k(2d-k+1)}$	Weak

Fig. 3. Comparison of repair-bandwidth and secure storage performance

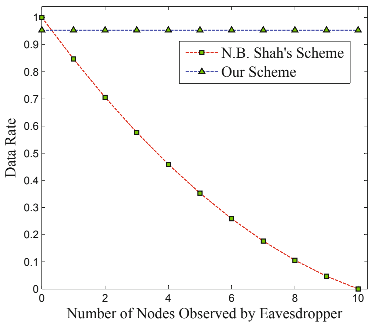

Fig. 4. An example with $(n, k, d) = (15, 10, 13)$

Next we will show that our scheme provides secure storage efficient(i.e., Higher Data Rate) guarantee relative to other similar work. [11–14] gave methods to construct schemes providing both the regenerating property and secure storage against eavesdroppers. But all these techniques provide *perfect-security* with expense of storage capacity so high, even unbearable. Taking N.B.Shah's work on secure MBR [13] for example without loss of generality, the random symbols amount to $[\ell d - \binom{\ell}{2}]\beta$ need to be mixed with the original symbols, and the data rate $\mathcal{R} = \frac{\mathcal{L}_{sec}}{\mathcal{L}} = \frac{[kd-\binom{k}{2}]\beta-[\ell d-\binom{\ell}{2}]\beta}{[kd-\binom{k}{2}]\beta} = 1 - \frac{\ell^2-(2d+1)\ell}{k^2-(2d+1)k}$ decrease rapidly when the intruder can eavesdropped more nodes for $\ell < k$. Fig.4 shows an example for $(n, k, d) = (15, 10, 13)$, we can see that the data rate is lower than 20%

and intolerable to us when $\ell \geq 7$. However, our scheme achieve a higher data rate (approximate 95% in Fig.4) and it does not vary with $\ell(< k)$ when the parameters (n, k, d) are fixed. Although this comes at the expense of reduced security, the *weak-security* is enough in practical scenario.

4.3 Computation Cost Analysis

Comparing with the PM-MBR scheme[3], our solution leads to additional computation cost for the purpose of security. Taking (n, k, d) as the design parameters, we could get the overhead of our scheme. The Preprocessing Procedure takes $O(d^2)$ Hash operations and $O(d^2)$ Xor operations in order to obtain the set P from the original symbol set S. The number of operations taken by the second and third procedure is equal to that of the original PM-MBR scheme. Especially in the Regeneration Procedure, where the Ψ_{rev} is chosen to be a Vandermonde matrix, $O(d) + O(d^2) + O(d \log d^2)$ arithmetic operations are needed to repair the failed node. The Reconstruction Procedure firstly perform the same recover action as conventional PM-MBR and then recover original symbols, as a inverse precess of the first procedure, *step 2* of this procedure takes the same amount of operations as the first procedure. Totally, our scheme takes $2O(d^2)$ Hash operations and $2O(d^2)$ Xor operations more than traditional PM-MBR codes, but this overhead is rather smaller than schemes exploit encryption mechanism.

5 Divergent Thinking

As is mentioned above, we utilize the secure hash function and PM-MBR code, and finally achieve the *weak security* guarantee with the loss of data capacity kept in a low level. As shown in Fig.3, data rate of our scheme is a constant(close to 100%) when the parameters (n, k, d) are fixed. A natural question arises: is it possible to achieve the storage security against ℓ-eavesdropper without loss of the data capacity? The answer is yes, now we briefly introduce a construct method below. Here the AONT(All-Or-Nothing Transform) [17] $\mathcal{T}$ is introduced first.

$\mathcal{T}$: $(X_1, X_2, ..., X_n) \rightarrow (Y_1, Y_2, ..., Y_n)$ with $X_i, Y_i (i = 1, 2, ..., n)$ drawn from a finite field $\mathbb{F}_q$. The work in [18] considers AONT and addresses unconditional security with respect to a single block of the original message. In other words, someone who have obtained the symbols $Y_1, Y_2, ..., Y_m$ is unable to invert the transform $\mathcal{T}$ and recover each X_i when $m < n$, if and only if $m = n$ the transform is invertible, this is an important property of the AONT.

Replace the preprocessing procedure in our scheme with an AONT, and take the original data symbols $s_1, s_2, ..., s_{L_s}$ as input of the AONT, then we can get result symbols $P = \{p_1, p_2, ..., p_{L_s}\}$. Next, fill the result symbols into the message matrix M. Denoting a subset of P as $\hat{P}$, and each symbol of $\hat{P}$ is placed into the first $k-1$ rows and first $k-1$ columns of the symmetric matrix M, thus $|\hat{P}| = \frac{k(2d-k+1)}{2} - (d - k + 1)$. Reviewing the conclusion of *Lemma 1*, we know that the intruder who eavesdrops $\ell < k$ nodes cannot obtain any information of the subset $P \setminus \hat{P}$. As a result, the eavesdropper is impossible to recover each single original symbol s_i because she did not obtain all the result symbols in $P = \{p_1, p_2, ..., p_{L_s}\}$, i.e., $I(s_i; C_E) = 0$. Obviously, without using the random symbols(as K in above scheme), we can also achieve a secure storage against ℓ-eavesdropper with $L_s = \mathcal{L}$ and thus $\mathcal{R} = 1$, i.e., without loss of data capacity.

We must recognize the fact that the security of both the two schemes, respectively utilizing secure hash function and AONT, relies on the perfect security of K and partial elements of P (these elements are placed in the position of massage matrix M except that in the first $k-1$ rows and first $k-1$ columns, we denote these elements as a set $\mathcal{Q}$). Thus, making full use of the set $\mathcal{Q}$, we believe that various schemes with good properties can be constructed. In the future, we will make an intensive research in this aspect.

6 Conclusion

In this paper, we proposed an efficient scheme which provides not only data reliability but also data security against passive ℓ-eavesdroppers in DSS, based on the Product-Matrix framework. To be viewed as a combination of minimum bandwidth regeneration code with secret sharing scheme, our scheme offers more advantages over previous work. Other than data security against eavesdropper, it satisfies both the MDS property and regeneration property (the repair-bandwidth is as low as the MBR code).

Considering the fact that similar repair efficient scheme all provide perfect security criterion for data stored in the system, however, this criterion is too strict to be practical, because it leads to an unaffordable loss of the storage capacity while ℓ get closer to k. In contrast, we adopt the weak security criterion instead, and give a definition of "Secure DSS against an ℓ-eavesdropper". Utilizing a secure hash function and Product-Matrix framework, our scheme finally achieves a high secrecy capacity and constant data rate close to 1 with fixed (n, k, d). The analysis result indicates that, our scheme is sufficiently secure, repair efficient, storage efficient and suitable for practical systems.

Furthermore, another approach was introduced, which combines the All-Or-Nothing Transform with PM-MBR code, thus achieving a secure storage against ℓ-eavesdropper without loss of data rate.

References

1. Dimakis, A.G., Godfrey, P.G., Wu, Y., Wainwright, M.J., Ramchandran, K.: Network Coding for Distributed Storage Systems. IEEE Trans. on Information Theory 56, 4539–4551 (2010)
2. Suh, C., Ramchandran, K.: Exact Regeneration Codes for Distributed Storage Repair Using Interference Alignment. In: Proc. IEEE International Symposium on Information Theory (ISIT), Austin (2010)
3. Rashmi, K., Shah, N.B., Kumar, P.V.: Optimal Exact-regenerating Codes for Distributed Storage at the MSR and MBR Points via a Product-Matrix Construction. IEEE Trans. on Information Theory 57(8), 5227–5239 (2011)

4. Tamo, I., Wang, Z., Bruck, J.: Zigzag Codes: MDS Array Codes with Optimal Rebuilding. IEEE Trans. on Information Theory 59, 1597–1616 (2013)
5. Oliveira, P.F., Lima, L., Vinhoza, T.T.V., Barros, J., Médard, M.: Coding for Trusted Storage in Untrusted Networks. IEEE Trans. on Information Forensics and Security 7(6) (2012)
6. Bloch, M., Barros, J.: Physical-Layer Security: From Information Theory to Security Engeering. Cambridge Univ. Press, Cambridge (2011)
7. Oliveira, P.F., Lima, L., Vinhoza, T.T.V., Médard, M., Barros, J.: Trusted Storage Over Untrusted Networks. In: Proc. IEEE Global Communications Conference (GLOBECOM2010), Miami, FL (2010)
8. Yamamoto, H.: Secret Sharing System Using (k, l, n) Threshold Scheme. Electronics and Communications in Japan (Part I: Communications) 69, 46–54 (1986)
9. Bessani, A., Correia, M., Quaresma, B., André, F., Sousa, P.: DepSky: Dependable and Secure Storage in a Cloud-of-Clouds. In: Proc. EuroSys 2011, Salzburg, Austria (2011)
10. Krawczyk, H.: Secret Sharing Made Short. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 136–146. Springer, Heidelberg (1994)
11. Pawar, S., El Rouayheb, S., Ramchandran, K.: Securing Dynamic Distributed Storage Systems Against Eavesdropping and Adversarial Attacks. IEEE Trans. on Information Theory 57(10), 6734–6753 (2012)
12. Rawat, A.S., Koyluoglu, O.O., Silberstein, N., Vishwanath, S.: Optimal Locally Repairable and Secure Codes for Distributed Storage Systems. In arXiv:1210.6954 (2013)
13. Shah, N.B., Rashmi, K.V., Kumar, P.V.: Information-Theoretically Secure Regenerating Codes for Distributed Storage. In: Proc. IEEE Global Communications Conference, GLOBECOM (2011)
14. Kurihara, M., Kuwakado, H.: Secret sharing Schemes Based on Minimum Bandwidth Regenerating Codes. In: 2012 International Symposium on Information Theory and its Applications (ISITA), pp. 255–259 (2012)
15. Bhattad, K., Narayanan, K.R.: Weakly Secure Network Coding. In: Proc. First Workshop on Network Coding, Theory, and Applicat. (NetCod), Riva del Garda, Italy (2005)
16. Shamir, A.: How to Share a Secret. Commun. ACM 22(11), 612–613 (1979)
17. Rivest, R.L.: All-or-Nothing Encryption and the Package Transform. In: Biham, E. (ed.) FSE 1997. LNCS, vol. 1267, pp. 210–218. Springer, Heidelberg (1997)
18. Stinson, D.R.: Something About All or Nothing (Transforms). Designs, Codes and Cryptography 22(2), 133–138 (2001)
19. Cui, T., Ho, T., Kliewer, J.: On Secure Network Coding Over Networks with Unequal Link Capacities and Restricted Wiretapping Sets. In: Proc. IEEE International Symposium on Information Theory, ISIT (2010)
20. Cai, N., Yeung, R.W.: Secure network coding. In: Proc. IEEE International Symposium on Information Theory (ISIT), Lausanne, Switzerland (2002)
21. El Rouayheb, S., Soljanin, E., Sprintson, A.: Secure network coding for wiretap networks of type II. IEEE Trans. on Information Theory 58(3), 1361–1371 (2012)
22. Luo, M.X., Yang, Y.X., et al.: Secure Network Coding Against Eavesdropper. Science In China Series F-Information Sciences 40(2), 371–380 (2010)
23. Adeli, M., Liu, H.: Secure Network Coding with Minimum Overhead Based on Hash Functions. IEEE Communications Letters 13(12), 956–958 (2009)
24. Dimakis, A.G., Ramchandran, K., Wu, Y., Suh, C.: A Survey on Network Codes for Distributed Storage. Proceedings of the IEEE 99(3) (2011)