An Efficient Dispersal and Encryption Scheme for Secure Distributed Information Storage*

Sung Jin Choi¹, Hee Yong Youn^{1**}, and Bo Kyoung Lee²

¹ School of Information and Communications Engineering
Sungkyunkwan University

440-746, Suwon, Korea +82-31-290-7949
{choisj, youn}@ece.skku.ac.kr

² Information Security Technology Division
Korea Information Security Agency
138-803, Seoul, Korea +82-2-405-5263
bkleee@kisa.or.kr

Abstract. Digital information is a critical resource, creating a need for distributed storage systems that provide sufficient data availability and security in the face of failures and malicious attacks. However, the requirements for achieving high availability and security with distributed storage systems usually conflict with each other. This paper proposes a novel data dispersal/encryption scheme improving both the availability and security of distributed storage system by using some important properties of matrices. It also allows complete recovery of original data even though they are partially damaged. Analysis shows it improves the availability about 15% compared with an efficient existing scheme for both read and write operation, while it allows secure storage simultaneously.

1 Introduction

Digital information has become a critical resource, creating a need for distributed storage systems that provide sufficient data availability and data security in the face of failures and malicious attacks. Many research groups have been exploring the design and implementation of such survivable storage systems. These systems are built on various decentralized storage system technologies sharing a same high level architecture. The challenge in the development of survivable storage is to achieve acceptable levels of performance and manageability. Moreover, a means for evaluating survivable storage systems is required to facilitate the design of the systems [1].

A data distribution scheme needs data encoding and partitioning algorithm. There exist many such algorithms applicable to survivable storage, including encryption, replication, striping, erasure-resilient coding, secret sharing, and various combinations of them. They offer different levels of performance (throughput), availability (prob-

* This work was supported by Korea Research Foundation Grant (KRF-2002-041-D00421)

** Corresponding author

ability that data can be accessed), and security (effort required to compromise the confidentiality or integrity of stored data).

For example, replication provides high availability at the sacrifice of network bandwidth and storage space, whereas short secret sharing provides high availability and security with low storage and network bandwidth overhead but at high computation overhead. Likewise, selecting the number of shares required to reconstruct a secret-shared value involves a trade-off between availability and confidentiality: if more machines must be compromised to steal the secret, then more machines must be operational to provide it legitimately. Availability of storage system needs to be improved through data distribution, and development of new efficient data distribution and protection scheme is inevitable.

In distributed storage system availability and security of data are greatly influenced by the policy how the data are dispersed. For example, complete recovery is possible even though most data are lost in one scheme while one node failure causes loss of entire data in another scheme, vice versa. Usually, the requirements for achieving high availability and security with distributed storage system conflict with each other, and there exist some tradeoff between them. This paper proposes a novel data dispersal/encryption scheme to improve both the availability and security of distributed storage system by using some important properties of matrices. It also allows complete recovery of original data even though they are partially damaged. Analysis shows that the availability is improved about 15% compared with an efficient existing scheme, while it allows secure storage.

The rest of the paper is organized as follows. The next section explains representative data distribution schemes. Section 3 presents the proposed dispersal/encryption scheme using eigenvalues and eigenvectors of matrices. In Section 4 we evaluate and compare the performance of the proposed scheme. Finally, Section 5 concludes the paper with future work.

2 Related Work

Secure distributed information storage system uses general threshold schemes to encode data before it is stored. Specifically, a p - m - n threshold scheme breaks data into n shares such that any m of the shares can reconstruct the original data and fewer than p shares reveal absolutely no information about the original data. Although encryption makes it difficult to ascertain the original data, it does not change the value of p since information is still available for theft. Different parameter selections of p - m - n expose a large space of encoding mechanisms for storage [2].

An example of a specific threshold scheme is N -way replication. It is a 1 - 1 - N threshold scheme. That is, each replica reveals some information on the encoded data (hence $p = 1$). A single replica is required to reconstruct the original data ($m = 1$), and there are N replicas to select from when attempting to reconstruct the original data ($n = N$). Table 1 lists some specific examples of general threshold schemes.

The threshold schemes can be implemented in different manners. Blakley's secret sharing scheme works in an m -dimensional space. Secrets (data) are points in the

Table 1. Specific threshold schemes.

Parameters	Description
$1-1-n$	Replication
$1-n-n$	Decimation (Striping)
$n-n-n$	Splitting (XORing)
$1-m-n$	Information Dispersal
$m-m-n$	Secret Sharing
$p-m-n$	Ramp Scheme

space, and the shares are multidimensional planes. Fewer than m shares represent a multidimensional plane that contains the secret. However, since all points in the field being considered are part of the plane, no information is revealed. With m shares, a single point of intersection - the secret - is determined. Figure 1 illustrates the Blakley’s secret sharing scheme. Shamir’s secret-sharing scheme is based on interpolating the coefficients of a polynomial by evaluating the polynomial at certain points [3].

Fig. 1. Blakley’s secret sharing scheme with $m=2, n=3$, and original data S .

All implementations of secret sharing use random numbers to guarantee that collecting fewer than p shares reveals no information on the original data. Rabin’s information dispersal algorithm, a $1-m-n$ threshold scheme, does not use random numbers. Indeed, the parameter p indicates the number of random numbers required per encoding (i.e. p -1 random numbers are required to encode a secret for any threshold scheme). Ramp schemes implement the full range of general $p-m-n$ threshold schemes, while they are like secret-sharing schemes up to p shares and information dispersal schemes with p or more shares [4-5].

Figure 2 shows some examples of threshold schemes. (a) Replication ($1-1-n$) increases availability by increasing the storage required by a factor of n . It provides no confidentiality because each share contains an entire copy of the original information object. (b) Splitting ($n-n-n$) increases confidentiality by increasing the storage required by a factor of n but decreases availability because all n shares must be available. (c) Decimation ($1-n-n$) divides information objects into n pieces and stores each piece separately. It decreases availability because all shares must be available while it offers no confidentiality because each share exposes $1/n$ of the original information. (d) Rabin’s information dispersal algorithm ($1-m-n$) offers a range of storage solutions that show trade-off between availability and required storage space. Like decimation, it does not offer confidentiality [6].

Fig. 2. Examples of simple threshold schemes.

The threshold schemes can be used instead of cryptographic techniques to guarantee the confidentiality, and multiple schemes can be combined. For example, short secret sharing encrypts the original information with a random key, stores the encryption key using secret sharing, and stores the encrypted information using secret sharing or information dispersal. Short secret sharing offers a different set of trade-offs between confidentiality and storage requirements than general threshold schemes. The confidentiality of the data stored by short secret sharing hinges on the difficulty of analyzing the information gained by collecting the shares because the information gained pertains to the encrypted data.

An extension to threshold schemes is cheater detection. Here the shares are constructed in such a fashion that a client reconstructing the original information object can tell, with high probability, whether some shares have been modified or not. This technique allows strong information-integrity guarantees. Cheater detection can also be implemented using cryptographic techniques such as adding digests to data before encoding or to the shares themselves after the data have been encoded [7].

Secret-sharing scheme is m - m - n threshold scheme that shows trade-off between confidentiality and availability; the higher the confidentiality guarantee, the more shares required to reconstruct the original information object. It can be thought of as a combination of splitting and replication techniques.

3 The Proposed Scheme

We first present some definitions and theorems required to explain the proposed dispersal/encryption scheme.

3.1 Eigenvalues and Eigenvector's Definition

• Definition 1.

Let A be an $n \times n$ matrix. A nonzero vector v is an eigenvector of A if Eq. (1) holds for some scalar λ . λ is called an eigenvalue of A corresponding to the eigenvector v . Eigenvalues are also known as characteristic, or proper, values or even as latent roots.

$$Av = \lambda v \quad (1)$$

Let us now discuss how to compute eigenvalues and eigenvector in general. Because

$$\begin{aligned}
Av = \lambda v &\Rightarrow Av = \lambda Iv \\
&\Rightarrow Av - \lambda Iv = 0 \\
&\Rightarrow (A - \lambda I)v = 0
\end{aligned} \tag{2}$$

We see that v is an eigenvector if and only if it is a nontrivial solution of the homogeneous system $(A - \lambda I)v = 0$. In this case v is a nonzero vector of the null space of $A - \lambda I$. The system has a nontrivial solution if and only if the determinant of the coefficient matrix is zero. Thus, λ is an eigenvalue of A if and only if $\det(A - \lambda I) = 0$. The following is a well known theorem [8].

• **Theorem 1.**

Let A be a square matrix.

1. A scalar λ is an eigenvalue of A if and only if $\det(A - \lambda I) = 0$.
2. A vector v is an eigenvector of A corresponding to an eigenvalue λ if and only if v is a nontrivial solution of the system $(A - \lambda I)v = 0$.

3.2 Dispersal/Encryption Scheme

We need a general method for finding eigenvalue and eigenvector by using Definition 1 and Theorem 1, and it is the Power Method. It computes the dominant eigenvalue and an eigenvector corresponding to the dominant eigenvalue. Without loss of generality, it is necessary to assume that A has the following two properties:

- i. There is a single eigenvalue of maximum modulus.
- ii. There is a linearly independent set of n eigenvectors.

According to the first assumption, the eigenvalues can be labeled such that

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \dots \geq |\lambda_n|. \tag{3}$$

According to the second assumption, there is a basis $\{u^{(1)}, u^{(2)}, \dots, u^{(n)}\}$ for C^n such that

$$Au^{(j)} = \lambda_j u^{(j)} \quad (1 \leq j \leq n). \tag{4}$$

Let $x^{(0)}$ be an element of C^n such that when $x^{(0)}$ is expressed as a linear combination of the basis elements $u^{(1)}, u^{(2)}, \dots, u^{(n)}$, the coefficient of $u^{(1)}$ is not 0. Thus,

$$x^{(0)} = a_1 u^{(1)} + a_2 u^{(2)} + \dots + a_n u^{(n)} \quad (a_1 \neq 0). \tag{5}$$

We form then $x^{(1)} = Ax^{(0)}$, $x^{(2)} = Ax^{(1)}$, ..., $x^{(k)} = Ax^{(k-1)}$ to have

$$x^{(k)} = A^k x^{(0)}. \tag{6}$$

In the following analysis there is no loss of generality in absorbing all the coefficients a_j in the vectors $u^{(j)}$ that they multiply. Hence, we may rewrite Eq. (5) as

$$\mathbf{x}^{(0)} = \mathbf{u}^{(1)} + \mathbf{u}^{(2)} + \cdots + \mathbf{u}^{(n)}. \quad (7)$$

By this equation and (6), we have

$$\mathbf{x}^{(k)} = \mathbf{A}^k \mathbf{u}^{(1)} + \mathbf{A}^k \mathbf{u}^{(2)} + \cdots + \mathbf{A}^k \mathbf{u}^{(n)}. \quad (8)$$

Using Eq. (4), we arrive at

$$\mathbf{x}^{(k)} = \lambda_1^k \mathbf{u}^{(1)} + \lambda_2^k \mathbf{u}^{(2)} + \cdots + \lambda_n^k \mathbf{u}^{(n)}. \quad (9)$$

$$\mathbf{x}^{(k)} = \lambda_1^k \left[\mathbf{u}^{(1)} + \left(\frac{\lambda_2}{\lambda_1} \right)^k \mathbf{u}^{(2)} + \cdots + \left(\frac{\lambda_n}{\lambda_1} \right)^k \mathbf{u}^{(n)} \right]. \quad (10)$$

Since $|\lambda_1| > |\lambda_j|$ for $2 \leq j \leq n$, we see that the coefficients $\left(\frac{\lambda_j}{\lambda_1} \right)^k$ tend to 0 and the

vector within the brackets converges to $\mathbf{u}^{(1)}$ as $k \rightarrow \infty$.

To simplify the notation, we write $\mathbf{x}^{(k)}$ in the form

$$\mathbf{x}^{(k)} = \lambda_1^k \left[\mathbf{u}^{(1)} + \boldsymbol{\varepsilon}^{(k)} \right]. \quad (11)$$

where $\boldsymbol{\varepsilon}^{(k)} \rightarrow 0$ as $k \rightarrow \infty$. In order to be able to take ratios, let φ be any linear functional on C^n for which $\varphi(\mathbf{u}^{(1)}) \neq 0$. Recall that a linear functional φ satisfies $\varphi(\alpha \mathbf{x} + \beta \mathbf{y}) = \alpha \varphi(\mathbf{x}) + \beta \varphi(\mathbf{y})$, for scalars α and β and vectors $\mathbf{x}$ and $\mathbf{y}$. (For example, φ could simply evaluate the j th component of any given vector.) Then

$$\varphi(\mathbf{x}^{(k)}) = \lambda_1^k \left[\varphi(\mathbf{u}^{(1)}) + \varphi(\boldsymbol{\varepsilon}^{(k)}) \right]. \quad (12)$$

Consequently, the following ratios converges to λ_1 as $k \rightarrow \infty$:

$$r_k \equiv \frac{\varphi(\mathbf{x}^{(k+1)})}{\varphi(\mathbf{x}^{(k)})} = \lambda_1 \left[\frac{\varphi(\mathbf{u}^{(1)}) + \varphi(\boldsymbol{\varepsilon}^{(k+1)})}{\varphi(\mathbf{u}^{(1)}) + \varphi(\boldsymbol{\varepsilon}^{(k)})} \right] \rightarrow \lambda_1. \quad (13)$$

This constitutes the Power Method for computing λ_1 . Since the direction of the vector $\mathbf{x}^{(k)}$ aligns more and more with $\mathbf{u}^{(1)}$ as $k \rightarrow \infty$, the method can also give us the eigenvector $\mathbf{u}^{(1)}$ [9]. If eigenvectors found are

$$\mathbf{u}^{(1)} = \begin{bmatrix} s_1 \\ \vdots \\ s_n \end{bmatrix} s_n \in \mathbf{R}, \mathbf{u}^{(2)} = \begin{bmatrix} t_1 \\ \vdots \\ t_n \end{bmatrix} t_n \in \mathbf{R}, \dots, \mathbf{u}^{(n)} = \begin{bmatrix} z_1 \\ \vdots \\ z_n \end{bmatrix} z_n \in \mathbf{R}. \quad (14)$$

Here P ($P = [u^{(1)} u^{(2)} \dots u^{(n)}]$) is actually stored data and matrix D consisting of eigen-

values becomes Decryption Key. For example, if original data is $A = \begin{bmatrix} 3 & 0 & 0 \\ -4 & 6 & 2 \\ 16 & -15 & -5 \end{bmatrix}$,

$\lambda_1 = 0, \lambda_2 = 1, \lambda_3 = 3$. Eigenvectors that correspond to each λ become $\begin{bmatrix} 0 \\ s \\ -3s \end{bmatrix}, \begin{bmatrix} 0 \\ 2t \\ -5t \end{bmatrix}, \begin{bmatrix} u \\ 0 \\ 2u \end{bmatrix}$ $s, t, u \in \mathbb{R}$. Therefore, if $s = t = u = 1$, the actually stored data

P and decryption key D are as follows.

$$P = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 2 & 0 \\ -3 & -5 & 2 \end{bmatrix}, D = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

An important property of the proposed scheme is that it allows both secret dispersal as a general threshold scheme and encryption of data at the same time. That is, one cannot extract original data even though the stored data P is available since deciding the original matrix using the eigenvector is NP-hard problem. Refer to the example above. There exist infinite ways to form P matrix by arbitrarily deciding the s, t , and u values. As a result, the proposed scheme offers high availability and security as well as complete recovery of original data even though some partial damage to the data occurs.

3.3 Data Recovery Scheme

We next show how the original data is recovered using the matrix diagonalization method.

• Definition 2.

Assume an $n \times n$ matrix A can be converted to a diagonal matrix D , which is called diagonalizable. Then there exists an invertible $n \times n$ matrix P such that

$$P^{-1}AP = D. \quad (15)$$

The process of finding matrices P and D is called diagonalization. First, it is worth to notice that if D is a diagonal matrix with diagonal entries $\lambda_1, \dots, \lambda_n$, then for $i=1, \dots, n$

$$De_i = \lambda e_i. \quad (16)$$

Hence, the standard basis vectors $e_1, \dots, e_n$ are eigenvectors of D . In particular, the eigenvectors of D are linearly independent. More generally, we have following Theorem 2.

• Theorem 2.

(Criterion for Diagonalization)

Let A be an $n \times n$ matrix.

1. A is diagonalizable if and only if it has n linearly independent eigenvectors.
2. If A is diagonalizable with $P^{-1}AP = D$, then the columns of P are eigenvectors of A and the diagonal entries of D are the corresponding eigenvalues.
3. If $\{v_1, \dots, v_n\}$ are linearly independent eigenvectors of A with corresponding eigenvalues $\lambda_1, \dots, \lambda_n$, then A can be diagonalized by

$$P = [v_1 v_2 \dots v_n] \text{ and } D = \begin{bmatrix} \lambda_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \lambda_n \end{bmatrix}. \quad (17)$$

• **PROOF.**

Let P be a matrix with columns of n-vectors $v_1, \dots, v_n$ and D be a diagonal matrix with diagonal entries $\lambda_1, \dots, \lambda_n$, respectively. Then

$$AP = A[v_1 v_2 \dots v_n] = [Av_1 Av_2 \dots Av_n]. \quad (18)$$

$$[\lambda_1 v_1 \lambda_2 v_2 \dots \lambda_n v_n] = [v_1 v_2 \dots v_n] \begin{bmatrix} \lambda_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \lambda_n \end{bmatrix} = PD. \quad (19)$$

If A is diagonalizable, with $P^{-1}AP = D$, then $AP = PD$. Hence, $Av_i = \lambda_i v_i$, $i=1, \dots, n$. So, the λ_i s are eigenvalues and the v_i s are corresponding eigenvectors.

Suppose that A has n linearly independent eigenvectors, say, $v_1 v_2 \dots v_n$ (the columns of P). If $\lambda_1, \dots, \lambda_n$ are the corresponding eigenvalues, then $Av_i = \lambda_i v_i$, $i=1, \dots, n$. If D is a diagonal matrix with diagonal entries $\lambda_1, \dots, \lambda_n$, then $AP = PD$ by (18) and (19). Because P is a square matrix with linearly independent columns, it is invertible. Hence, $P^{-1}AP = D$ and A is diagonalizable [10].

$$P^{-1}AP = D. \quad (20)$$

If we multiply P matrix to both sides in upside form, Eq. (20) becomes $PP^{-1}APP^{-1} = PDP^{-1}$. Since $PP^{-1} = I$, $P^{-1}P = I$ (I: identity matrix),

$$A = PDP^{-1} \\ = [u^{(1)} \dots u^{(n)}] \begin{bmatrix} \lambda_1 & \dots & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \dots & \dots & \lambda_n \end{bmatrix} [u^{(1)} \dots u^{(n)}]^{-1}. \quad (21)$$

P: Set of Eigenvectors, D: Decryption Key consisting of eigenvalues.

Therefore, the original data can be recovered from P that was stored in the storage nodes by $A = PDP^{-1}$. For the same example of Section 3.2, we get original data by

$$\begin{bmatrix} 0 & 0 & 1 \\ 1 & 2 & 0 \\ -3 & -5 & 2 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} 4 & -5 & -2 \\ -2 & 3 & 1 \\ 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 3 & 0 & 0 \\ -4 & 6 & 2 \\ 16 & -15 & -5 \end{bmatrix}$$

4 Performance Evaluation

The substantial body of prior work in building highly-available systems provides a clear metric for evaluating information availability: the probability that a desired piece of information can be accessed at any given point in time. Assuming uncorrelated failures, this probability can be computed from the probabilities that required system components are available. For example, a p - m - n threshold scheme requires at least m of the n storage nodes containing the shares to be operational to perform a read. If f_{node} is the probability that a storage node has failed or is otherwise unavailable, then the availability of the stored information is

$$\text{Availability}_{\text{read}} = \sum_{i=0}^{n-m} \binom{n}{i} (f_{\text{node}})^i (1-f_{\text{node}})^{n-i}. \quad (22)$$

For writes, the computation of availability depends upon the system's design. A system could require that all of n specific nodes be operational for a write to succeed. This approach clearly has poor availability. To improve write availability in a system with $N > n$ storage nodes, the write operation can attempt to write the shares at different storage nodes until it has completed n writes. Alternatively, a system could allow a write to complete when fewer than n shares have been written. This requires more work during failure recovery or reduces read availability of the stored data (because n is effectively lower for that data). We calculate write availability using the last approach. Thus, m storage nodes of the N storage nodes in the system must be available for a write to be performed, and write availability is [11]

$$\text{Availability}_{\text{write}} = \sum_{i=0}^{N-m} \binom{N}{i} (f_{\text{node}})^i (1-f_{\text{node}})^{N-i}. \quad (23)$$

Fig. 3. Comparison of read availability

Fig. 4. Comparison of write availability

Figure 3 and 4 show the comparison of the read and write availability of the proposed scheme along with striping and ramp scheme (p -2- n). Here up to 100 nodes were

tested. The figures reveal that the proposed scheme display incomparably better availability than striping while it is consistently better than the ramp scheme for about 15% for both read and write operation.

5 Conclusion and Future Work

In this paper we have presented a new data dispersal/encryption scheme to improve the availability and security of distributed storage system at the same time by using some important properties of matrix. As a result, the proposed scheme offers high availability and security as well as complete recovery of original data even though partial damage occurs to the data. Analysis shows that the availability is improved about 15% compared with an existing representative scheme for both read and write operation, while it allows secure storage.

Necessity of storage system providing safe data storage is inevitable as dependency on storage device is embossed at this point. Therefore more efforts are required to develop effective data distribution and encryption techniques customized to various applications and operation environment. A formal design methodology which identifies the tradeoff between the performance measures such as availability, security, and operation speed also needs to be developed.

References

1. Yvo Desmedt: Some Recent Research Aspects of Threshold Cryptography: Information Security, First International Workshop ISW '97, Lecture Notes in Computer Science #1196, M. Mambo E. Okamoto, E. Davida, Ed., (1997) 158–173
2. A. Shamir: How to Share a Secret: Comm. ACM. (1979) 612–613
3. E. Karnin, J. Greene, M. Hellman.: On Secret Sharing Systems: IEEE Trans. Information Theory (1983) 35–41
4. G. R. Blakley, Catherine Meadows: Security of ramp schemes: Advances in Cryptology, Springer-Verlag, (1985) 242–268
5. A. De Santis and B. Masucci: Multiple Ramp Schemes: IEEE Trans. Information Theory (1999) 1720–1728
6. Jay J. Wylie, Michael W. Bigrigg, John D. Strunk, Gregory R. Ganger, Han Kiliccote, Pradeep K. Khosla: Survivable Information Storage systems: IEEE Computer (2000)
7. Y. Deswarte, L. Blain, J. Fabre: Intrusion Tolerance In Distributed Computing Systems: Proc. IEEE Symp. Security and P privacy, IEEE CS Press, Los Alamitos, Calif (1991) 110–120
8. George Nakos, David Joyner: Linear Algebra with Applications, Brooks/Cole USA (1998) 442–455
9. David Kincaid, Ward Cheney: Numerical Analysis Mathematics of Scientific Computing. 2nd edn, Brooks/Cole USA (1996) 274–276
10. Birkhauser: Linear Algebra, Birkhauser Boston (1997) 216–219
11. Mehmet Bakaloglu, Jay J. Wylie, Chenxi Wang, Gregory R. Ganger: On Correlated Failures in Survivable Storage Systems: School of Computer Science Carnegie Mellon University, Pittsburgh, PA15213 (2002)