EEO: an Efficient MDS-Like RAID-6 Code for Parallel Implementation

Jun Feng, Yu Chen*, Douglas Summerville

Dept. of Electrical and Computer Engineering, SUNY – Binghamton, Binghamton, NY 13902

***Abstract* – In recent years, researchers have found that some XOR erasure codes lead to higher performance and better throughput in fault-tolerant distributed data storage applications. However, little consideration has been given to the advantages of parallel processing or hardware implementations taking advantage of the emergence of multi-core processors. This paper presents an efficient horizontal MDS-like (Maximum Distance Separable) RAID-6 scheme, called EEO, which significantly improves the performance of the decoding procedure in parallel implementations with little storage overhead. We show that EEO is the fastest and most efficient double disk failure recovering algorithm in RAID-6 at the cost of only two more parity symbols. In practice, it is very useful for application where high decoding throughput is desired.**

***Keywords: XOR, RAID-6, Fault-tolerant, Storage System*.**

1. Introduction

In modern storage systems, RAID (Redundant Array of Independent Disks) techniques are known as the preferable technique to achieve high performance and reliability. Among the well known RAID techniques, RAID-6, which can tolerate two failure-disks, has the best balance between storage efficiency and reliability. Erasure-coding technologies can provide both high fault tolerance and high storage efficiency [4].

While all the erasure coding techniques are feasible in practice, coding schemes based on the Reed-Solomon (RS) code are most popular with their MDS (Maximum Distance Separable) property. The information dispersal algorithm [9] or the Parchive adopted in some schemes or systems [6] indeed are derived from the RS codes. To date, several classes of horizontal MDS array codes have been successfully designed to simultaneously recover double storage node failure including the EVENODD code [1], [2] X code [11], RDP (Row-Diagonal Parity) scheme [3], Libertine Code [7] or their derivative schemes [5], [10].

Although X-code [11] is an elegant two-erasure code, it is not a RAID code since it is a vertical code and does not fit the RAID-6 specification of having coding devices *P* and *Q*, where *P* is a simple parity device [7]. Actually, all non-MDS codes and vertical codes are not implementable in RAID-6 systems [7]. A recent examination of the performance of the codes for RAID-6 using Classic Reed-Solomon codes and Cauchy Reed-Solomon codes based on Open-Source Erasure Coding Libraries concluded that special-purpose RAID-6 codes vastly outperform their general-purpose counterparts and RDP performs the best of these by a narrow margin [8].

In practice, data symbols are more important than parity symbols and users typically care more about how fast failed data symbols can be recovered in the decoding scheme than how fast the parity symbols can be constructed. With the emergence of the multi-core processor and programmable hardware implementation, the performance of the decoding algorithm can be improved significantly if some of operations can be processed in parallel. Most of the codes for RAID-6, such as EVENODD and RDP, can start two iterative decoding procedures when there is only one failed symbol in the chain with slope 1. However, a faster decoding procedure is still expected.

The reason that codes based on XOR operation can start two decoding chains simultaneously is that one of the coding slopes includes an imaginary symbol to generate the parity, which leads to only one real symbol and one imaginary symbol erased on a specific chain and is used to start the chain. This leads to the question of whether speedup can be achieved in the parallel decoding process if both the coding slopes have the imaginary symbol.

In this paper, we propose the EEO (Extension of the EVENODD code) code scheme, which is an efficient MDS array-like scheme with optimal decoding at the cost of adding only two parity symbols on the parity devices. The EEO code can be effectively implemented in parallel to accelerate the decoding procedure using multiple threads. Any two failed columns can be recovered. It is not strictly a MDS code due to the two extra redundant symbols. However, it possesses the similar properties of the MDS code when applied to RAID-6.

The remainder of this paper is organized as follows. Section 2 provides the algebraic description of the EEO coding and decoding procedures. We also prove that the EEO scheme can correct any two erasures. Section 3 gives a performance analysis and a comparison with EVENODD and RDP codes. Section 4 concludes the paper.

2. EEO scheme

The EEO scheme consists of (n-1) row × (n+2) column

* Manuscript submitted on Nov. 18, 2009 to *the 33rd IEEE Sarnoff Symposium 2010*. Corresponding author: Yu Chen, E-mail: ychen@binghamton.edu, Tel.: (607) 777-6133, Fax: (607) 777-4464.

+ 2 symbols, where the first (n-1) rows × n columns contain information symbols and the last 2 columns contain parity symbols. Unlike other codes, the EEO scheme forbids the row parity and uses the extra two parity symbols to eliminate the generation and regeneration of S adjustor. The most important issue is that the two extra parity symbols accelerate the parallel decoding procedure dramatically, which will be described in Section 3.

A. Encoding Procedure

Figure 1 illustrates the encoding procedure of the EEO. In Fig. 1(a), the first (n-1) rows × n columns contain the information symbols represented by '$d_{i,j}$', and the last 2 columns contain the parity symbols represented by '$c_{i,0}$', '$c_{i,1}$'. The last row in the first n columns represents the imaginary area that is used for comprehension. In Fig. 1(a), $d_{4,0}$, $d_{4,1}$, $d_{4,2}$, $d_{4,3}$, and $d_{4,4}$ are imaginary symbols. Figures 1(b) and 1(c) show that the same symbols construct the chains with columns $c_{i,0}$ and $c_{i,1}$ respectively. The encoding algorithm of parity columns is represented as the following:

$$c_{i,0} = \bigoplus_{t=0}^{n-2} d_{t,t-i}, \quad (1)$$

and

$$c_{i,1} = \bigoplus_{t=0}^{n-2} d_{t,i-t}. \quad (2)$$

where i = 0, 1, ……, n-2.

$d_{0,0}$	$d_{0,1}$	$d_{0,2}$	$d_{0,3}$	$d_{0,4}$	$c_{0,0}$	$c_{0,1}$
$d_{1,0}$	$d_{1,1}$	$d_{1,2}$	$d_{1,3}$	$d_{1,4}$	$c_{1,1}$	$c_{1,1}$
$d_{2,0}$	$d_{2,1}$	$d_{2,2}$	$d_{2,3}$	$d_{2,4}$	$c_{2,0}$	$c_{2,1}$
$d_{3,0}$	$d_{3,1}$	$d_{3,2}$	$d_{3,3}$	$d_{3,4}$	$c_{3,0}$	$c_{3,1}$
$d_{4,0}$	$d_{4,1}$	$d_{4,2}$	$d_{4,3}$	$d_{4,4}$	$c_{4,0}$	$c_{4,1}$

(a) An example of n=5

♦	♣	♥	♠	●	♦	
●	♦	♣	♥	♠	●	
♠	●	♦	♣	♥	♠	
♥	♠	●	♦	♣	♥	
♣	♥	♠	●	♦	♣	
$d_{i,0}$	$d_{i,1}$	$d_{i,2}$	$d_{i,3}$	$d_{i,4}$	$c_{i,0}$	$c_{i,1}$

(b) Slope = -1, construction of parity column $c_{i,0}$.

♦	♣	♥	♠	●		♦
♣	♥	♠	●	♦		♣
♥	♠	●	♦	♣		♥
♠	●	♦	♣	♥		♠
●	♦	♣	♥	♠		●
$d_{i,0}$	$d_{i,1}$	$d_{i,2}$	$d_{i,3}$	$d_{i,4}$	$c_{i,0}$	$c_{i,1}$

(c) Slope=1, construction of parity column $c_{i,1}$.

Figure 1. Illustration of the EEO encoding procedure.

B. EEO decoding description

This section focuses on proving how to recover two failed data columns since it is clear how to recover single data column. That is, the i-th and j-th column failed where $0 \le i < j \le n-1$.

Lemma 1: *Assume that there are two columns i and j, where $0 \le i < j \le n-1$, The iterative chain starting from imaginary point (n-1, i) or (n-1, j) with slope -1 and then with slope 1 and -1 alternatively can traverse all the points in i-th and j-th columns until it reaches point (n-1,j)if and only if n is a prime number.*

Proof:

First, we assume that n is a prime number. The imaginary row is included for convenience. Suppose the starting point is (n-1,i), the next point by slope -1 is (n-1-(j-i), j), then the point (n-1-2(j-i), i) by slope 1. Iteratively, the sequence pair (n-1-k×(2×(j-i)+1), j), (n-1-k×2×(j-i), i) where $0 \le k \le n-1$ can be derived.

The sequence has 2×n points, and each column has been hit n times. The sequence points in i-th column are (n-1-k×2×(j-i), i). The sequence points in j-th column are (n-1-k×(2×(j-i)+1), j). Then proving the lemma is equivalent to proving that the sequence can traverse the i-th and j-th columns respectively.

Consider the i-th column, for example. The sequence can be simplified to (r-k×c), where r=n-1, c = 2×(j-i), where $0 \le k \le n-1$. The proof of the lemma in i-th column is equivalent to proving that the sequence (r-k×c) does not repeat and it occurs only once where $0 < c$, $r < n$.

Suppose there are two integers x, y where $0 \le x < y \le n-1$ and make sequence (r- x×c) = (r- y×c). That is, there are two equal sequence numbers, and (x-y)×c mod n = 0. As n is a prime number, c cannot be divided by n, then (x-y) should be divisible by n, which is contradicted with the fact that $0 \le x < y \le n-1$, and n is a prime number. Therefore, the sequence cannot repeat when $0 \le k \le n-1$, the iterative chain traverses the n symbols in the i-th column. Since (n-1-n×c, i) mod n = (n-1, i), the next point hits the starting point.

On the other hand, if n is not a prime number n can be factored into two factors: n_1 and n_2, where $n_1 < n$, and $n_2 < n$. Considering (x-y)×c mod n = 0, if we make $c = n_1$ or n_2, then (x-y) is not required to be dividable by n and the equation is still valid, which implies that the sequence (r-k×c) could repeat when $0 < c < n$. There must be at least one symbol that cannot be reached by the iterative chain. Then n is a prime number is a necessary condition.

The sequence in the j-th column can be proved to traverse the n points in a similar manner. Then the lemma from (n-1,i) is proved. The starting point from (n-1, j) can be also be proven in a similar way.

Lemma 1 is proven.

According to Lemma 1, from any starting point, the data symbols in two columns can be hit one by one. The two failed columns can be recovered from any starting point. Just like other XOR erasure codes, it is very simple to implement the iterative operation in software or hardware. It is obvious that we can start recovery chains in parallel.

However, in RDP code, one of them stops quickly when it hit the missing chain, which leads to more steps to recover all lost symbols. Then before starting with four chains in EEO there are two questions to be answered:

- *Question #1*: Can the four iterative chains encounter in the middle and when?
- *Question #2*: What the advantage of starting with four chains?

Here, '*encounter*' means that the next point the chain reaches has already been hit by another chain. Before deriving the conclusion, several lemmas are first proved.

Lemma 2. *For two sequences, seq1 = k×m mod n, seq2 = k×(n-m) mod n, where 1≤k≤n-1, m= j-i, n is a prime number, seq2 is the reverse of sequence 1, and* $seq1_k + seq2_k = n$, $seq1_k \neq seq2_k$.

Proof:

For *seq1*, $seq1_k = x = k \times m \bmod n$, that is,

$$k \times m = x + w \times n \tag{3}$$

Similarly, $seq2_k = y = k \times (n-m) \bmod n$, that is,

$$(k \times n - k \times m) \bmod n = y + h \times n \tag{4}$$

where $0<x, y<n$, and w, h, k are all integers.

Substituting equation (3) into equation (4), we get $k \times n - x - w \times n = y + h \times n$, that is:

$$x + y = (k - w - h) \times n \tag{5}$$

where $0<x, y<n$, $0<x + y<2 \times n$, k-w-h is an integer.

Then we get $x + y = n$. That is, $seq1_k + seq2_k = n$.

The reverse of sequence 1 is:

$$Seq1_{n-k} = z = (n-k) \times m \bmod n \tag{6}$$

and $$Seq2_k = y = (k \times n - k \times m) \bmod n \tag{7}$$

Since $k \times n \bmod n = n \times m \bmod n = 0$, by comparing equations (6) and (7), $Seq1_{n-k} = -k \times m = Seq2_k$. Therefore, sequence 2 is just the reverse of sequence 1. Without considering the order, *seq1 = seq2*. If considering the order, we know $seq1_k + seq2_k = n$, the sequence is not repeated and n is odd number, it is easier to find that $seq1_k \neq seq2_k$.

Lemma2 is proven.

Example #1:

Suppose $n = 5$, $m = 2$, n-$m = 3$, then starting from 0, *seq1* is {2, 4, 1, 3}; and *seq2* is {3, 1, 4, 2};

Lemma 3. *For two sequences, seq1 = k×m mod n, seq2 = k×(n-m) mod n, where 1≤k≤n-1, m= j-i, n is a prime number, if they run with the same speed each sequence will cover the first half of the entire data set when they encounter. And the union of the first half of the seq1 and seq2 is equal to seq1 or seq2 without repeating.*

Proof:

According to Lemma 2, sequence 2 is merely the reverse of sequence 1. That is, $seq2_k = seq1_{(n-k)}$. Then the traces of sequences 1 and 2 can be seen as the trace of two points in one sequence moving towards each other from the opposite end. Since the number of sequence 1 or 2 is n-1, which is even and the speed is the same, then both points can cover the equal distance (the half of the sequence) when they encounter.

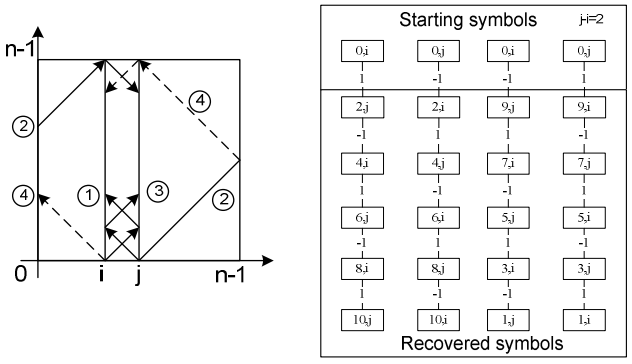
Assume the first halves of sequence 1 and sequence 2 are $seq1_k$ and $seq2_k$, where $1 \le k \le (n-1)/2$. Then $n-(n-1)/2 = (n+1)/2 \le n-k \le n-1$, since sequence 2 is just the reverse of sequence 1. That is, $seq2_k = seq1_{(n-k)}$, the first half of $seq2_k$ can be equal to the second half of $seq1_{n-k}$. Thus, $\{seq2_k, \mid 1 \le k \le (n-1)/2\} = \{seq1_m \mid (n+1)/2 \le n \le (n-1)\}$, therefore the union of the sequence set $seq = \{seq1_k \cup seq2_k, \mid 1 \le k \le (n-1)/2\} = \{seq1_k \mid 1 \le k \le (n-1)/2\} \cup \{seq1_m \mid (n+1)/2 \le n \le (n-1)\} = \{seq1_k \mid 1 \le k \le (n-1)\}$.

Lemma 2 indicates that $seq1_k + seq2_k = n$ and n is odd number, it is easier to find that $seq1_k \neq seq2_k$. If there are k repeating symbols in the first halves of *seq1* and *seq2*, the size of the union is $(n-1)/2 + (n-1)/2 - k = (n-1)-k$, which contradicts the conclusion above. That is, the first halves of *seq1* and *seq2* have no repetition.

Lemma 3 is proven.

Now, we can give the following theorem.

Theorem 1. *Assume there are two columns i and j, where 0≤i<j≤n-1, The four iterative chains starting from (n-1, i) and (n-1, j) with slope 1 and slope -1 respectively and then with slope -1 and 1 alternatively in the same speed can each traverse 1/4 of the lost symbols when they encounter. The union of four sequence chains traverses all the failed symbols and each of them traverse 1/4 of the failed symbols. The decoding speed increases 4 times.*



(a) Four chains **(b) Decoding example**

Figure 2. Illustration of the EEO decoding procedure

Proof:

Since $(n-m) \bmod n = (-m) \bmod n$, and there is no difference when starting the original point from the left-bottom point or right-top point. For the sake of convenience, we will use them in the proof.

The four iterative chains can be illustrated in Fig. 2(a), chain 1 and chain 2 starts from slope 1 and chain 3 and chain 4 starts from slope -1. Through calculation, the four chains sequences are:

1. $\{(2mk, i), ((2k+1)m, j), \ldots\}$,
2. $\{(-2(k+1)m, i), -2mk, j), \ldots\}$,
3. $\{(2mk, j), ((2k+1)m, i), \ldots\}$,
4. $\{(-2(k+1)m, j), -2mk, i), \ldots\}$,

Where $0 \leq k \leq (n-1)$

Then the proof of Theorem 1 can be reformatted as the following question: *can the four chains cover the two failed columns*? According to Lemma 1, neither of the chains can traverse all the symbols in the two columns; they must encounter somewhere before hitting the imaginary row.

Meanwhile, in each step, chain 1 and chain 3, chain 2 and chain 4 have the same row with different columns. In other words, in each step chain 1 and chain 3 will never hit the same symbol and likewise with chain 2 and chain 4. Then we can simplify the procedure by mapping the four chains into two sequences that iteratively traverse one column (i.e. the i-th column):

1. $\{(2mk), ((2k+1)m), \ldots\}$; and
2. $\{(-2(k+1)m), -2mk), \ldots\}$,

The two sequences can be further reduced to $\{mk\}$ and $\{-mk\}$, respectively.

The two iterative chain sequences are with the step m and $(n-m)$ respectively, where $m = j-i$. According to Lemma 3, the union of two sequence chains traverses all the failed symbols and each of them traverses half of them.

Then Theorem 1 is proved. Theorem 1 provides the answer to the *Question #1* in section 3.

As for the *Question #2*, it always only takes $(n-1)/2$ steps to recover the failed symbols, since there is only two recover chains. It is the optimal decoding process.

Based on Theorem 1, the EEO decoding algorithm can be described as follows:

Decoding Algorithm:

Step 1. Set $k = n-1$, set the starting point with (k, i) & (k, j);

Step 2. Start the four chain sequences to find the next symbols;

Step 3. If the next symbol is not on the imaginary row or is not hit, Go to Step2; Else exit.

It is easier to illustrate the operations of the algorithm through an example. Assume that column i and j have failed where $j-i=2$, $n=11$, the decoding procedure is illustrated in Fig. 2(b). According to Theorem 1, the iterative decoding process will continue until the four chains encounter in the middle and each of them traverses 1/4 of the failed symbols.

C. RAID-6 Properties

Theorem2: *EEO scheme is a RAID-6 scheme if and only if n is a prime number.*

Proof:

First, by the coding scheme, we can see that all the parity symbols are stored in columns P and Q. Next we will prove that any two of the failed columns can be recovered.

According to Lemma 1, from any starting point, the data symbols in two columns can be hit one by one. Therefore, the two failed columns can be recovered.

Theorem 2 is proven.

3. Performance Analysis

Table 1 compares the properties of the EEO code and two other codes, EVENODD and RDP. The reason for choosing these for comparison is that these two codes are the typical codes for RAID-6. Based on Table 1, if only XOR operations are considered, EEO does not possess much advantage over the other two codes. However, in practice, since information symbols are more important than the parity symbols, users typically care more about how fast the failed data symbols can be recovered in the decoding scheme than how fast the parity symbols can be constructed. The performance of reconstruction does not only depend on the XORs, but also on how many reconstructions can be conducted in parallel. This will reduce the time consumption dramatically.

Table 1 Normal Property Comparison

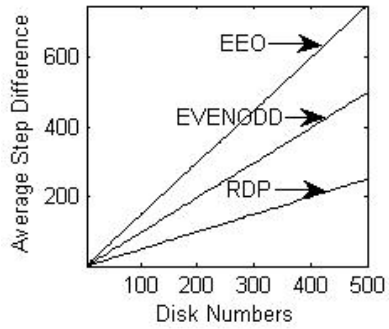
	Coding	Decoding	Update	Storage
EVEN ODD	$2n^2-2n-1$	$2n^2+2n-5$	>2	$n/(n+2)$
RDP	$2n^2-6n+4$	$2n^2-6n+4$	2	$(n-1)/(n+1)$
EEO	$2n^2-2n$	$2n^2-4n+2$	2	$n/((n+2)+(\frac{2}{n-1}))$

Although all the XOR erasure codes can be efficiently implemented using hardware and/or software, the performance of the decoding procedure can be improved dramatically if it can be processed in parallel. For EVENODD, while one step is needed to recover S, in practice, it is not preferable to be implemented to recover all S_u in n parallel procedures in one step. Thus, $(n-1)$ steps with two parallel chains are needed to retrieve the failed symbols iteratively since one of the slopes is 0 and the row parity chain cannot be started with two lost symbols. Therefore, ideally $(n+1)$ steps are needed to recover the failed symbols.

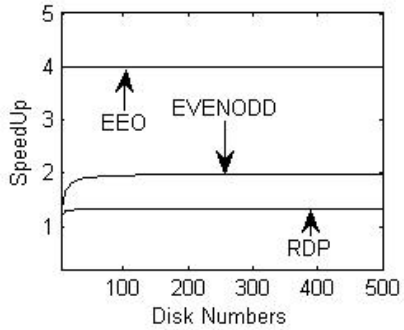
RDP can only be processed in a single chain most of the time since the failed symbols can only be recovered one by one using the iterative algorithm. Even if it starts with two chains, one of them will stop quickly when it hits the missing chain. In contrast, the analysis in Section 3 has shown that our EEO code can start four chains in parallel and only $(n-1)/2$ steps are needed to recover the failed symbols.

Figure 3(a) shows the average number of steps of the three code schemes with the same combination of two-failed columns (C_{n-1}^2), where the number of average

steps is defined as the total number of maximum steps in each group chain. For example, for RDP, when n = 5, there are six possible group chains, the two steps in each of them are (0,8), (0,8), (0,8) , (2,6), (2,6), (4,4), the maximum step number in each is 8, 8, 8, 6, 6, 4. The average step number is (8+8+8+6+6+4)/6= 40/6=6.67, and 2×(n-1) = 8 steps are needed to recover all the failed symbols in one chain. Then the difference is (8-6.67) = 1.33. In Fig. 3(a), the difference among the number of average steps increases linearly with the growth of the disk number. The more the disks are adopted, the better the parallel performance of EEO.



(a) Step Difference



(b) Speed Up

Figure 3. Parallel Performance Comparison.

Figure 3(b) shows the comparison of speedup among the three codes. The speedup is defined as the ratio of the number of steps needed to recover all the failed symbols in one chain to the total maximum number of steps in each XOR codes. For example, in RDP, when n = 5, the SpeedUp = 6×2×(5-1)/(8+8+8+6+6+4) = 48/40=1.2. That is, the speed can be improved by 120%. Fig. 3(b) also shows that speedup of the EEO code is stable at 400%, the EVENODD can be close to 200% and RDP is less than 150% in the simulation experiment using 500 disks based on parallel processing.

4. Conclusion

In this paper, we propose the EEO code, a new horizontal scheme that achieves the fastest decoding time among the reported erasure codes. The EEO code is actually an extension of the EVENODD code, and requires only (n-1)/2 iterative recovery steps in its decoding operations due to its four parallel iterative chains. The simplicity and flexibility of EEO make it easy to be implemented in hardware or parallel computing environments. Detailed proof has been presented, and a comparison study of its performance has been carried out.

In particular, the efficient parallel erasure decoding algorithm for the EEO code has been discussed thoroughly. The analysis of the parallel decoding procedure shows that the EEO code has achieved the highest efficiency in the decoding process compared with other horizontal MDS erasure codes for RAID-6. Although it requires two more sectors, the EEO code is very suitable for high availability in practical data storage systems.

References

[1] M. Blaum, J. Brady, J. Bruck, and J. Menon. "EVENODD: An Efficient Scheme for Tolerating Double Disk Failures in RAID Architectures," *IEEE Transactions on Computers* (44:2), 1995, pp. 192-202.

[2] M. Blaum, P. Farrell, and H. van Tilborg, "Array Codes," *Handbook of Coding Theory*, V.S. Pless and W.C. Huffman, eds., Elsevier Science Publishers B.V., 1998.

[3] P. Corbett, B. English, A. Goel, T. Grcanac, S. Kleiman, J. Leong, and S.Sankar, "Row-Diagonal Parity for Double Disk Failure Correction", *Proc. of USENIX FAST 2004*, Mar. 31 to Apr. 2, San Francisco, CA, USA.

[4] J. Feng, Y. Chen, and D. Summerville, "A Survey on the Application of the XOR Erasure Code for Distributed Storage System," submitted to *the IEEE Communications Surveys and Tutorial*, Feb. 2010, under review.

[5] C. Jin, H. Jiang, D. Feng and L. Tian, "P-Code: A New RAID-6 code with optimal properties," *23rd International Conference on Supercomputing*, New York, June, 2009.

[6] J. S. Plank, "A Tutorial on Reed-Solomon Coding for Fault-Tolerance in RAID-like Systems," *Software: Practice and Experience*, vol. 27, no.9, pp. 995–1012, Jan. 1999.

[7] J. S. Plank, "The RAID-6 Liberation Codes," *the 6th USENIX Conference on File and Storage Technologies* (FAST'08), pp. 97–110, San Jose, CA, Feb., 2008.

[8] J. S. Plank, J. Luo, C. Schuman, L. Xu, and Z. Wilcox-O'Hearn, "A Performance Evaluation and Examination of Open-Source Erasure Coding Library for Storage," *the 7th USENIX Conference on File and Storage Technologies* (FAST'09), San Francisco, CA, February, 2009.

[9] M. Rabin, "Efficient Dispersal of Information for Security, Load Balancing and Fault Tolerance", *J. ACM*, 32(4), 335-348, Apr. 1989.

[10] G. Wang, X. Liu, S. Lin, G. Xie and J. Liu, "Generalizing RDP Codes Using the Combintorial Method," *the 7th IEEE International Symposium on Network Computing Applications* (NCA-08), Cambridge, MA, 2008.

[11] L. Xu and J. Bruck. "X-Code: MDS Array Codes with Optimal Encoding," *IEEE Trans. on Information Theory* (45), 1999, pp. 272--276.