제 18 회 한국정보처리학회 추계학술발표대회 논문집 제 9 권 제 2 호 (2002.11)

안전한 분산정보저장을 위한 효율적인 분산/암호화 기법

최성진*, 윤희용*, 이보경**, 최중섭**, 박창원***, 이형수***
*성균관대학교 정보통신공학부
**한국정보보호진흥원 정보보호기술팀
***전자부품연구원 정보시스템 연구센터
e-mail : *{Choisj, youn}@ece.skku.ac.kr, **{bklee, jschoi}@kisa.or.kr,
***{parkcw, hslee}@keti.re.kr

An Efficient Dispersal/Encryption Scheme for Secure Distributed Information Storage

Sung Jin Choi*, Hee Yong Youn*, Bo Kyoung Lee**,
Joong Sup Choi**, Park ChangWon***, Lee Hyung Su ***
*School of Information and Communications, Sung Kyun Kwan University
**Information Security Technology Division, Korea Information Security Agency
*** IT System Research Center, Korea Electronics Technology Institute

요 약

인터넷 사용량의 증가, 전자상거래의 활성화 그리고 저장장치의 발전과 더불어 많은 사람들이 디지털 정보를 편리하게 이용할 수 있게 되었다. 이에 따라 저장장치에 대한 의존도, 보안성 그리고 생존성에 대한 요구사항 또한 매우 높아져가고 있는 실정이다. 따라서, 본 논문에서는 분산정보저장시스템의 생존성을 높이기 위해 필수적으로 필요한 행렬의 고유값과 고유벡터를 이용한 새로운 데이터 분산/암호화기법을 제안하고, 제안된 기법의 가용성을 평가한다. 제안된 기법은 데이터의 분할과 암호화를 동시에 허락하여 보안성을 높임과 동시에 기존의 기법과 비교하여 15%정도의 가용성 향상을 보인다.

1. 서론

인터넷 사용량의 증가, 전자상거래의 활성화 그리고 저장장치의 발전과 더불어 많은 사람들이 디지털 정보를 편리하게 이용할 수 있게 되었다. 이에 따라 저장장치에 대한 의존도, 보안성 그리고 생존성에 대한 요구사항이 매우 높아져가고 있는 실정이다. 특히, 기업 및 개인정보의 보호와 예상치 못한 상황에서의 안전한 데이터 보전을 위해서는 높은 생존성을 가지는 저장장치시스템의 필요성이 요구되고 있다. 이러한 저장장치의 생존성을 높이기 위해서는 분산저장기법을 통하여 저장시스템의 가용성을 높이는 것이 선행되어야 하며, 이를 위한 새로운 분산 및 데이터 보호기법의 연구개발이 절실히 필요하다.

분산저장시스템에서는 정보를 분산된 각 저장노드 단위로 저장한다. 이 때 데이터의 가용성은 데이터를 분산시키는 기법 및 정책에 따라 크게 변하게 되는데, 즉, 분산하는 알고리즘에 따라 거의 대부분의 데이터를 잃어도 완벽한 복구가 가능한 반면, 특정 수 이상의 데이터를 잃으면 복구가 불가능한 경우도 있다. 특히, 정보를 단순 분할 저장하거나 보안성을 높이기 위해 분할 저장하는 두 경우 모두 부분적인 데이터의 손실 및 파괴가 전체 데이터의 손실로 이어지는 경우가 발생하게 된다. 따라서, 본 논문에서는 분산정보저장시스템의 생존성을 높이기 위해 필수적으로 필요한 행렬의 고유값과 고유벡터를 이용한 새로운 데이터 분산/암호화기법을 제안하고, 제안된 기법의 가용성을 평가한다. 제안된 기법은 데이터의 분할과 암호화를 동시에 허락하여 보안성을 높임과 동시에 기존의 기법과 비교하여 15%정도의 가용성 향상을 보인다.

본 논문의 구성은 다음과 같다. 2 장에서는 기존에 대표적으로 사용되는 threshold 기법을 이용한 데이터 분산기법에 대하여 알아보고, 3 장에서는 본 논문에서 제안하는 분산/암호화기법을 소개하고, 제안된 기법의 수식적 해석을 통해 가용성을 평가하고자 한다. 마지막으로 4 장에서는 결론 및 향후 연구과제를 제시한다.

2. 관련 연구

안전한 분산정보저장시스템에서 일반적인 threshold 기법에 의거한 데이터 분산기법은 크게 복제(Replication), 스트라이핑 (Striping), 정보 분산 (Information Dispersal), 스플리팅(Splitting)등으로 나뉘어 진다.

복제는 한 개의 데이터를 n 개의 완전한 복사본으로 만들어 분산 저장하는 기법으로 전체 노드 중 단지 하나만 살아 남아도 원래의 데이터를 완전하게 복구 시킬 수 있다. 이 기법의 단점은 침입자가 단지 하나의 노드에 접근하는 것만으로 모든 자료가 누출된다는 것이다. 스트라이핑은 하나의 데이터를 n 개의 조각으로 분할시켜서 저장하는 기법으로 1 개의 데이터를 저장하기 위해 1 개의 저장 공간만을 이용한다. 이 기법의 단점은 단 하나의 노드 손실만으로도 원래의 데이터를 완벽하게 복구시키지 못한다는 점이다. 정보 분산은 스트라이핑과 같이 데이터를 분할한 후에 복사본을 만들어 분산 저장하는 기법으로 일부 노드에 손실이 생겨도 원래의 데이터를 복구할 수 있다. 이 기법의 단점은 침입자가 일부분의 데이터를 얻더라도 완전하지는 않지만 어느 정도의 의미 있는 정보를 알 수 있다는 것이다. 스플리팅은 n-1 개의 저장 노드에는 임의의 난수를 발생시켜 저장하고 나머지 하나의 노드에는 발생시킨 n-1 개의 난수 들과 원래 데이터 값을 XOR 시켜 저장하는 기법으로 하나의 노드에 대한 침입은 전혀 의미가 없다. 이 기법의 단점은 하나의 노드 손실만으로도 원래 데이터의 복구가 불가능하다는 점이다 [1].

현재 분산 정보 저장 시스템을 위한 대표적인 분산 기법은 램프(Ramp) 기법과 Blakley 의 보안 분배가 있다 [2]. 램프 기법은 p-m-n threshold 기법으로 나타내는 여러 분산 기법들을 포괄적으로 적용 가능하게 하는 기법으로, p 는 최소로 유효한 단위 데이터의 묶음의 갯수를, m 은 원본 데이터로의 완전한 복구를 위해서 최소로 필요한 분할 단위 데이터 묶음의 갯수를, n 은 저장 되어 지는 저장장치 노드의 갯수를 의미한다 [3]. Blakley 의 보안 분배기법은 정보를 m 차원의 공간에서의 한 점으로 나타내는데, m-1 차 다항식 m 개 이상의 교점으로써 표현한다. 각 저장 노드에는 하나의 다항식에 대한 정보만 있으므로 정보의 누출이 어렵게 되고, 정보의 재구성을 위해서는 최소한 m-1 개의 노드가 살아있기만 하면 된다 [4].

3. 제안하는 행렬의 고유값과 고유벡터를 이용한 분산/암호화 기법

3.1 고유값과 고유벡터의 정의

□ **정의 1**

A 를 n×n 행렬이라고 하면, 스칼라 λ 에 대해서 Av=λv 식을 만족하는 0 이 아닌 벡터 v 를 고유벡터라고 한다. 이때 스칼라 λ 를 고유벡터에 대응하는 고유값이라고 한다.

□ **정리 1**

A 를 정방 행렬이라고 하면,

1. 스칼라 λ 가 고유값이라고 하면, $\det(A-\lambda I)=0$ 이 된다.
2. 벡터 v 가 고유값 λ 에 대응하는 고유벡터라고 하면, 동차(homogeneous) 시스템 $(A-\lambda I)=0$ 는 0 이 아닌 해(nontrivial solution)를 갖게 된다.

다음은 n 차의 정사각행렬 A 의 고유값을 구하는 일반적인 방법을 알아본다. λ 를 A 의 고유값이라 하면, $Ax=\lambda x \Leftrightarrow Ax=\lambda I_n x \Leftrightarrow (\lambda I_n - A)x=0$ 이 된다. $x \neq 0$ 이므로 동차연립방정식 $(\lambda I_n - A)x=0$ 은 0 이 아닌 해를 가져야 한다. 따라서 $\det(\lambda I_n - A)=0$ 에서 나오는 특성방정식을 풀면 고유값 λ 를 구할 수 있다 [5].

3.2 분산/암호화 기법

이제 우리가 제안하는 고유값과 고유벡터를 이용한 분산/암호화 기법에 대해서 설명하겠다. 우선, 정의 1 에 의해 고유값을 구하기 위해 데이터를 행렬로 변환한다. A 가 변환된 행렬이라고 하면, 다음과 같이 표현할 수 있다.

$$A=\begin{bmatrix} a_{11} & \cdots & \cdots & \cdots & a_{1n} \\ \vdots & a_{22} & & & \vdots \\ \vdots & & \ddots & & \vdots \\ \vdots & & & \ddots & \vdots \\ a_{n1} & \cdots & \cdots & \cdots & a_{nn} \end{bmatrix}$$

v 가 0 이 아닌 벡터라면 정의 1 의 식에 의해 다음 식이 성립한다.

$$\begin{bmatrix} a_{11} & \cdots & \cdots & \cdots & a_{1n} \\ \vdots & a_{22} & & & \vdots \\ \vdots & & \ddots & & \vdots \\ \vdots & & & \ddots & \vdots \\ a_{n1} & \cdots & \cdots & \cdots & a_{nn} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_n \end{bmatrix}=\lambda\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_n \end{bmatrix}$$

위의 식을 정리하면 다음과 같은 식이 나온다.

$$(a_{11}-\lambda)x_1+a_{12}x_2+\cdots\cdots+a_{1n}x_n=0$$
$$a_{21}x_1+(a_{12}-\lambda)x_2+\cdots\cdots+a_{2n}x_n=0$$
$$\vdots$$
$$a_{n1}x_1+a_{n2}x_2+\cdots\cdots+(a_{nn}-\lambda)x_n=0$$

이 방정식을 다시 행렬로 변환하면,

$$\begin{bmatrix} a_{11}-\lambda & \cdots & \cdots & \cdots & a_{1n} \\ \vdots & a_{22}-\lambda & & & \vdots \\ \vdots & & \ddots & & \vdots \\ \vdots & & & \ddots & \vdots \\ a_{n1} & \cdots & \cdots & \cdots & a_{nn}-\lambda \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_n \end{bmatrix}=\begin{bmatrix} 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix}.$$

이 된다. 정리 1.2 와 $v \neq 0$ 이라는 조건에 의해 A-λI 의 행렬식(determinant)은 0 이 되어야 한다. A-λI 의 행렬식을 구하기 위해서는 다음에 나오는 정의를 살펴보아야 한다.

□ **정의 2**

자연수의 집합 S={1,2,...,n}의 순열(permutation)이란 S 에서 S 로의 일대일 대응 σ 이다. 즉, 집합의 원소인 자연수 1,2,…,n 을 빠지거나 중복됨이 없이 어떤 순서로 나열한 것이다. S 의 순열 $j_1 j_2 \cdots j_n$ 에서 큰 자연수가 작은 자연수보다 먼저 나타나면 이 순열은 반전

(inversion)을 가졌다고 한다. 또, 순열이 가진 반전의 총 갯수가 짝수이면 이 순열은 짝치환, 홀수이면 홀치환이라고 한다.

□ **정의 3**

함수 $\mathrm{sgn}: S_n \rightarrow \{+1,-1\}$ 을 다음과 같이 정의한다.

$$\mathrm{sgn} = \begin{cases} +1 & (\sigma\text{:짝순열}) \\ -1 & (\sigma\text{ 홀순열}) \end{cases}$$

□ **정의 4**

행렬 $A=[a_{ij}]$가 n 차의 정사각행렬일 때, A 의 행렬식을 det(A) 또는 |A|로 나타내고 다음과 같이 정의한다.

$$\det(A) = \sum_{\sigma \in S_n} \mathrm{sgn}(\sigma) a_{1\sigma(1)} a_{2\sigma(2)} \cdots\cdots a_{n\sigma(n)}$$

이렇게 정의 2, 3 그리고 4 를 이용하여 $\lambda(\lambda_1,\lambda_2,...,\lambda_n)$ 값을 구하게 되는데, 이 λ를 고유값이라고 한다. 그러면, 다음에 나오는 벡터를 각 λ에 대응하는 고유벡터라고 하면 다음과 같이 나타낼 수 있다.

$$v_1=\begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ \vdots \\ s_n \end{bmatrix} s_n \in R,\ v_2=\begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ \vdots \\ t_n \end{bmatrix} t_n \in R,\ \cdots\cdots v_n=\begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ \vdots \\ z_n \end{bmatrix} z_n \in R$$

위와 같이 나온 n 개의 고유벡터가 실제로 저장되는 데이터이고, 고유값으로 이루어진 행렬 D 는 복호화 키가 된다.

□ **예제 1**

원래의 데이터를 $\begin{bmatrix} 3 & 0 & 0 \\ -4 & 6 & 2 \\ 16 & -15 & -5 \end{bmatrix}$ 라고 하면, 정의 1, 2, 3, 4 와 정리 1 에 의해 $\lambda_1=0,\ \lambda_2=1,\ \lambda_3=3$ 이 되고, 각 λ에 대응하는 고유벡터는 $\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} \begin{bmatrix} 0 \\ 2t \\ -5t \end{bmatrix} \begin{bmatrix} u \\ 0 \\ 2u \end{bmatrix}$ s,t,u ∈ R

이 된다. 따라서, s=u=t=1 로 했을 때, 실제로 저장되는 데이터 P 와 복호화 키 D 는 다음과 같다.

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

이 기법의 특징은 데이터의 분할과 암호화가 동시에 된다는 것으로서 일반적인 threshold 기법의 비밀분산 조건을 만족하고 있다. 즉, 고유벡터(분할된 데이터)를 가지고서 원래의 행렬(데이터)을 찾는 것은 NP Hard 문제이기 때문에 n 개의 분할된 데이터를 얻게 되어도 원래의 데이터를 알아 낼 수가 없다. 그리고 각 고유벡터들의 원소들은 실수에 대응하는 수의 비로 이루어졌기 때문에, 무한대의 조합으로 표현될 수 있다. 따라서 데이터의 부분적인 손실이나 파괴가 일어난다고 해도 원래의 데이터를 완벽하게 복구 할 수 있을 뿐만 아니라 높은 가용성과 보안성을 갖는다.

3.3 행렬의 대각화(diagonalization)를 이용한 데이터의 재 복구

우리가 제안한 고유값과 고유벡터를 이용한 분산/암호화 기법에 의해 분산 저장된 데이터를 재 복구하기 위해서는 다음에 나오는 정의와 정리를 살펴보아야 한다.

□ **정의 5**

A 가 어떤 대각선 행렬과 닮은 행렬일 때, 즉 적당한 가역행렬 P 가 존재하여 $P^{-1}AP$가 대각선 행렬일 때 A 를 대각화 가능한 행렬이라 하며 행렬 P 는 A 를 대각화하는 행렬이라고 한다.

□ **정리 2**

n 차의 정사각행렬 A 가 대각화 가능할 필요충분 조건은 A 가 n 차의 일차독립인 고유벡터를 갖는 것이다. 이 때, A 는 자신의 고유값을 주 대각선성분으로 갖는 대각선행렬 D 와 닮은 행렬이다.

□ **증명 2**

(충분조건): A 가 대각화 가능하면 적당한 가역행렬 $P=[p^{(1)}p^{(2)}\cdots p^{(n)}]$ 에 대하여 $P^{-1}AP=B$ 인 대각선행렬 $B=\mathrm{diag}\{b_1,...,b_n\}$ 이 존재한다. 이 때, $AP=PB$ 이므로 $Ap^{(1)}=b_1p^{(1)},...,Ap^{(n)}=b_np^{(n)}$ 이 된다. 따라서 $b_1,...,b_n$ 은 A 의 고유값이고 B=D 이다. 그러므로, $Ap^{(1)}=\lambda_1p^{(1)}, Ap^{(2)}=\lambda_2p^{(2)},...,Ap^{(n)}=\lambda_np^{(n)} \rightarrow (1)$. 그런데, P 가 가역이므로 $p^{(1)},...,p^{(n)}$ 은 일차 독립이고, (1)에 의해 고유값 $\lambda_1,...,\lambda_n$ 에 각각 대응하는 A 의 고유벡터이다.

(필요조건): A 의 고유값 $\lambda_1,...,\lambda_n$에 대응하는 일차독립인 고유벡터를 $p^{(1)},p^{(2)},...,p^{(n)}$ 이라 하고 이것을 열벡터로 갖는 행렬을 $P=[p^{(1)},p^{(2)},...,p^{(n)}]$라 하자. 그러면,

$$\begin{aligned} AP&=[Ap^{(1)}Ap^{(2)}...Ap^{(n)}] \\ &=[\lambda_1p^{(1)}\lambda_2p^{(2)}..\lambda_np^{(n)}] \\ &=[p^{(1)}p^{(2)}...p^{(n)}]\begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \cdots & 0 \\ 0 & 0 & \cdots & 0 \end{bmatrix} \\ &=PD \end{aligned}$$

이 된다. 그런데 P 의 열벡터들은 일차독립이므로 P 는 가역이다. 따라서 $P^{-1}AP=D$ 이고 A 는 대각화 가능하다 [6].

3.2 장에서는 행렬의 고유값과 고유벡터를 이용한 분산/암호화 기법에 대해서 살펴 보았다. 다음에는 정의 5 와 정리 2 를 이용하여 원래의 데이터를 복구하는 방법을 살펴 본다.

데이터의 재 복구를 위해서는 증명 2 의 마지막에 쓰인 다음 식을 자세히 살펴 볼 필요가 있다.

$$P^{-1}AP=D$$

위 식에서 P 는 가역 행렬이므로, 양변에 P 행렬을 곱해주면 $PP^{-1}APP^{-1}=PDP^{-1}$ 이 된다. 그리고 $PP^{-1}=I$, $P^{-1}P=I$ (I:항등행렬) 이기 때문에 다음식이 성립한다. 밑의 식에서 P 는 고유벡터들의 집합을 나타내고 D 는

복호화 키를 나타낸다.

$$A = PDP^{-1}$$
$$= [v_1 \cdots v_n] \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix} [v_1 \cdots v_n]^{-1}$$

따라서, 실제로 저장노드에 저장된 데이터 P 를 이용하여, $A = PDP^{-1}$ 식에 의해 원래의 데이터를 완전하게 복구 할 수 있다.

□ 예제 2

예제 1 에서 구한 P 를 이용하여 P^{-1} 를 구한다음, P 와 D, P^{-1} 을 $A = PDP^{-1}$ 에 대입하면 다음과 같이 원래의 데이터를 구할 수 있다.

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

3.4 가용성(Availability) 평가

가용성이란 자료를 저장해 놓은 저장노드 중 일부에 이상이 생겼을 때 (특정 노드지역(Pool)의 정전, 디스크의 파괴, 시스템의 다운 등과 같은 경우) 남아있는 노드만을 이용해서도 원래의 자료를 복구해낼 수 있는 특성을 말한다.

다음은 가용성 평가를 위해 사용된 식으로 m 은 원본데이터로의 완전한 복구를 위해서 최소로 필요한 분할 단위 데이터 묶음의 개수를 나타내며, n 은 저장될 저장장치 노드의 개수를 나타낸다. 마지막으로 F_{node} 는 각 노드가 실패할 확률을 나타낸다 [7].

$$\text{Availability} = \sum_{i=0}^{n-m} \binom{n}{i} \times F_{node}{}^{i} \times (1 - F_{node})^{n-i}$$

그림 1 은 스트라이핑, 램프 기법의 가용성과 제안된 분산/암호화 기법의 가용성을 비교한 것이다.

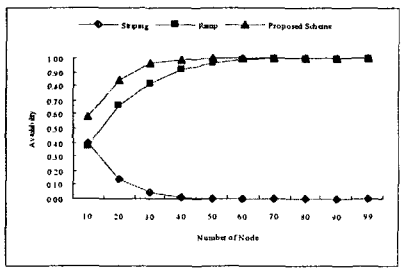

그림 1. 가용성 평가.

그림 1 의 그래프를 보면 제안된 분산/암호화 기법이 기존의 램프 기법과 비교하여 가용성에 있어서 대략 15%정도의 성능 향상을 보이고 있는 사실을 알 수 있다.

그림 2 는 가용성이 m 에 따라 변화하는 상태를 보여주는데, m 이 작을수록 가용성이 높아진다는 사실을 알 수 있다. 따라서 m 의 값이 작으면 사고가 일어나도 적은 수의 조각만으로도 원래 데이터를 재구성할 수 있다는 장점이 있다. 반대로 m 의 값이 증가하면, 침입자를 찾아내기가 용이하고 데이터의 중복이 줄게 되어 전제 저장장치 사용량이 줄어들게 되지만, 데이터를 읽고 쓰는데 필요로 하는 CPU, 네트워크 같은 리소스의 양이 증대하게 되고, 만일 사고가 일어났을 경우 살아남은 노드만으로는 전체를 복구할 수 없는 경우가 일어나게 된다. 따라서, 응용/정책에 따라 시스템의 목적에 적합한 최적의 파라미터를 절충하여 결정하는 것이 중요하다.

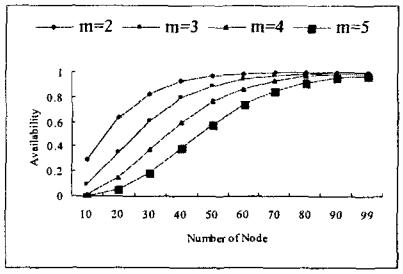

그림 2. m 에 대한 가용성 변화.

4. 결론 및 향후과제

본 논문에서 제안한 분산/암호화 기법은 데이터의 분할과 암호화를 동시에 하고, 기존의 기법과 비교하여 가용성 측면에서 대략 15%정도의 가용성 향상을 보인다.

저장장치에 대한 의존도가 높아지면서 안전한 데이터 보관이 가능한 분산저장시스템의 필요가 부각되고 있는 이 시점에서, 새로운 분산 기법과 데이터 보호기법을 위해 많은 노력이 필요할 것으로 본다. 따라서, 향후 연구 과제로 이미 제안된 분산기법보다 가용성과 보안성이 높은 분산/암호화 기법을 개발하는 것이 목표이다.

참고문헌

[1] Jay J.Wylie Michael W.Bigrigg, John D.Strunk, "Survivable Information Storage System," University of Carnegie Mellon, August 2000.

[2] De Santis, A.; Masucci, B. "Multiple Ramp Schemes," Information Theory, IEEE Transactions on, July 1999.

[3] G. R. Blakley and C. Meadows, "Security of Ramp Schemes,"*in advances in Cryptology'Lecture Notes in Computer Science*, Berlin, 1985.

[4] B. Blakley, G. Blakley, A.H. Chan and J. Massey, "Secret Sharing Schemes with Disenrollment", Proceedings of Crypto92, Springer Lecture Notes in Computer Science, pages 540 – 548, 1998.

[5] Birkhauser. "Linear Algebra," pages 209-216, 1997.

[6] George Nakos, David Joyner. "Linear Algebra with Applications," pages 441-467, 1998.

[7] Mehmet Bakkaloglu, Jay J. Wylie, Chenxi Wang, Gregory R. Gager, "On Correlated Failures in Survivable Storage Systems," University of Carnegie Mellon, May 2002.