# 서바이벌 스토리지 시스템을 위한 최적 정보 분할 기법

송 성 근[†] · 윤 희 용[††] · 이 형 수[†††] · 이 강 신[††††]


## 요 약

오늘날 정보 시스템에서 정보의 가용성(Availability), 무결성(Integrity), 기밀성(Confidentiality) 등을 지원하는 것이 중요하다. 서바이벌 스토리지 시스템(Survivable Storage System)은 악의적인 공격이나 시스템의 고장으로부터 데이터의 가용성, 무결성, 기밀성 등을 보장하기 위해 여러 스토리지 서버(Storage Server)나 데이터 베이스(Data Base)에 암호화하여 분산 저장시킨다. 정보 분할 기법(Information Dispersal Scheme)은 오버헤드(Overhead)가 크지 않으면서 높은 가용성과 보안성(Security)을 보장하는 기법들 중에 하나이다. 본 논문에서는 IDS(Information Dispersal Scheme)들의 집합이 주어졌을 때, 가용성 측면에서 가장 최적인 IDS를 결정하는 알고리즘을 제안한다. 서버의 수, 스토리지 용량, 오퍼레이션(Operation) 속도 등 여러 요소들이 IDS를 결정하는데 있어서 제약으로 작용하기 때문에 최적인 IDS를 찾기란 매우 어렵다. 따라서 이 알고리즘은 가용하고 안전한 스토리지 시스템을 디자인하는데 매우 유용할 것이다.


# Optimal Information Dispersal Scheme for Survivable Storage Systems

Sung Keun Song[†] · Hee Yong Youn[††] · Hyung Soo Lee[†††] · Kang Shin Lee[††††]


## ABSTRACT

Supporting the availability, integrity, and confidentiality of the information is crucial. The survivability storage systems require to encode and distribute data over multiple storage nodes or data base to survive failures and malicious attacks. Information dispersal scheme is one of the most efficient schemes allowing high availability and security with reasonable overhead. In this paper, we propose an algorithm determining the optimal $(m, n)$-IDS in terms of availability, given a set of IDS's. The proposed algorithm will be very useful for designing a highly available and secure storage system since many factors such as node number, storage space, operation speed, etc. interact with each other and thereby finding an optimal information dispersal scheme is very difficult.


**키워드 : 가용성(Availability), IDS(Information Dispersal Scheme), 정보 확장율(Information Expansion Ratio), 임계 확률(Critical Probability)**

## 1. 서 론

오늘날 인터넷, 전자상거래, 네트워크 등 IT 분야의 발달과 급격한 사용 증가로 우리 사회가 점점 디지털 정보에 의존되면서, 정보 시스템의 요소들 중 정보의 가용성, 무결성, 기밀성 등을 지원하는 것이 중요하게 되었다. 특히 시스템이 관리해야 할 중요 정보의 양이 급격히 증가하면서 저장장치시스템의 생존성(Survivable)과 정보의 가용성이 가장 중요한 사항으로 대두되었다. 여기서 가용성이란 인증된 사용자가 시스템 구성 요소의 고장(Failure) 상태와 상관없이 수시로 정보를 얻을 수 있는 가능성을 의미한다. 일반적으로 사용자들은 중요한 정보를 안전하게 저장할 수 있고, 항상 액세스(Access)가 가능하며 기밀성 및 무결성이 보장되는 안전한 시스템을 필요로 한다. 서바이벌 스토리지 시스템은 그런 종류의 시스템이라 할 수 있다. 서바이벌 스토리지 시스템의 종류로는 파시스(PASIS), 에스포(S4), 인터메모리(Intermemory), 퍼블리어스(Publius), 이볼트(e-Vault) 등이 있으며 이에 대한 연구가 활발히 이루어지고 있다.

서바이벌 스토리지 시스템[1]은 악의적인 공격이나 시스

※ 이 논문은 2003년도 두뇌한국21사업과 한국학술진흥재단(KRF-2002-041-D00421), 한국정보보호진흥원의 침입감내기술개발(2003-S-402)의 지원에 의하여 연구되었음.
† 준 회 원 : 성균관대학교 대학원 전기전자컴퓨터공학부
†† 종신회원 : 성균관대학교 정보통신공학부 교수
††† 정 회 원 : 전자부품연구원 정보시스템연구센터 책임연구원
†††† 정 회 원 : 한국정보보호진흥원 기반보호기술팀 팀장
논문접수 : 2003년 1월 24일, 심사완료 : 2003년 10월 24일

템 요소들의 고장으로부터 데이터의 가용성, 무결성, 기밀성 등을 보장하기 위해 여러 스토리지 서버나 데이터 베이스 등에 암호화하여 분산 저장시킨다. 그러기 위해 서바이벌 스토리지 시스템은 데이터의 가용성 및 보안성이 보장되는 데이터 분산 기법들을 이용한다. 대부분의 시스템들은 데이터의 보안성을 증가시키기 위해 데이터 분산 기법에 별도의 암호화 프로그램을 이용하고 있다. 데이터 분산 기법들은 여러 가지가 있는데, 대표적으로 복제(Replication), 스플리팅(Splitting), 데시메이션(Decimation), 정보 분할(Information Dispersal)[2], 비밀 분할(Secret Sharing)[3], 램프(Ramp)[4] 등이 있다. 이 기법들 중 어떤 데이터 분산 기법을 쓰느냐에 따라 서바이벌 스토리지 시스템에서 다른 가용성, 보안성, 성능(Performance)등이 나타난다. 데이터 분산 기법들 중 IDS은 오버헤드가 크지 않으면서 높은 가용성을 보장하는 기법들 중에 하나이다. 이 기법 자체로는 높은 수준의 데이터 보안성을 보장은 못하지만, 다른 암호화 프로그램과 연동이 용이하다는 장점이 있다. 본 논문에서는 이러한 장점들을 고려하여 IDS에 초점을 맞추어 연구한다.

IDS는 원본 데이터의 $m$개의 조각(Piece)을 가용성 개선을 위해 $n$개로 복제하여 서버에 분산 저장하는 기법이다. 원본 데이터로 재구성할 때는 $m$개의 데이터 조각이 필요하다. 즉, IDS는 일부 서버의 고장에도 불구하고 $m$개의 데이터 조각만 가용하면 원본 데이터를 재구성할 수 있다. 다시 말하면 IDS을 사용하는 시스템은 일부 서버의 고장을 묵인할 수 있다. IDS는 데이터의 가용성과 보안성을 증가시킬 뿐만 아니라 작업량을 서버간에 분배한다. 지금까지 IDS을 이용한 많은 예들이 제안되었는데 그러한 예들을 살펴보면, 시스템에서 서버의 수가 제한되어 있으면 최적인 서버 고장 묵인 능력(Fault-tolerant capability)을 갖게 하는 IDS의 $m$, $n$값을 결정하는게 중요하다는 것을 알 수 있다.

본 논문에서는 $m$과 $n$의 관계에 대해서 연구하고, 그 결과로 IDS들의 집합이 주어졌을 때, 가용성 관점에서 최적인 IDS를 찾는 알고리즘을 제안한다. 서버의 수, 스토리지 용량, 오퍼레이션 속도 등 여러 요소들이 IDS를 결정하는데 있어서 제약으로 작용하기 때문에 최적인 IDS를 찾기란 매우 어렵다. 따라서 이 알고리즘은 가용하고 안전한 스토리지 시스템을 디자인하는데 매우 유용할 것이다.

본 논문의 구성은 다음과 같다. 2장에서는 IDS의 기본적인 특성들에 대해서 설명한다. 3장에서는 IDS들을 비교할 때, 어떠한 IDS가 최적인 IDS인지 설명하고, 4장에서는 IDS 집합들이 주어졌을 때 가용성 관점에서 최적인 IDS를 찾는 알고리즘을 제안한다. 마지막으로 5장에서 결론을 맺는다.

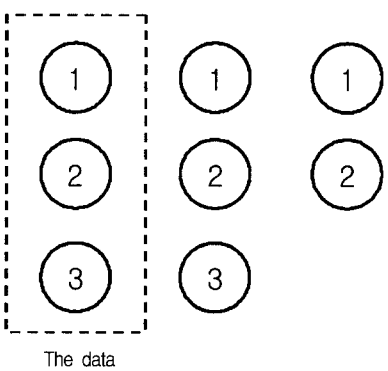
## 2. IDS의 기본적인 특성

이 장에서는 IDS의 기본적인 특성에 대해 설명한다. 먼저 이해를 돕기 위해 이 논문에서 사용된 표기법에 대해서 설명한 다음 $m$과 $n$의 관계에 대해서 설명한다. 그 다음 고전적인(Classic) 가용성 공식의 문제점을 증명하고 보완된 새로운 가용성 공식을 제안한다. 그리고 최적인 IDS를 선택하는 어려움을 설명한다.

### 2.1 표기법

- $n$ : 복제에 의한 스토리지 서버에 분산 저장되는 데이터 조각의 총 개수
- $m$ : 원본 데이터를 재구성하는데 필요한 데이터 조각의 개수
- $k$ : $n/m$ ; 정보 확장율(IER : Information Expansion Ratio) ; $k \geq 1$
- $P_s$ : 서버의 성공확률(Server probability) ; $0 < P_s < 1$
- $P_d(m, n)$ : $(m, n)$-IDS의 가용성 ;
- $P_d^*((i,j),(m,n))$ : 임계 확률(Critical Probability) ;
  $P_d(i,j) = P_d(m,n)$인 $P_s$
- $Class_i$ : IER이 $i$인 모든 IDS들의 집합
- $(m,n)_{(i,j)}$-IDS : 바운드(bound) IDS ; $Class_a$의 $(m, n)$-IDS는 $Class_b$의 $(i,j)$-IDS를 기점으로 $Class_b$의 IDS들과 임계 확률을 갖거나 갖지 않는다.

### 2.2 $m$과 $n$의 관계




(그림 1) (3, 8)-IDS의 데이터 조각

지금까지 위에서 설명했듯이 IDS는 원본 데이터의 $m$개의 조각을 가용성 개선을 위해 $n$개로 복제하여 서버에 분산 저장하는 기법이다. 원본 데이터로 재구성할 때는 $m$개의 데이터 조각이 필요하다. 그러므로 $m$과 $n$의 관계는 다음과 같다.

$$n = km \ (k \geq 1, 1 \leq m \leq n)$$

IDS의 IER인 $k$가 1보다 작으면, 데이터 손실이 발생하기 때문에 데이터 조각으로 원래의 데이터를 재구성하기가 불가능하다. 그러므로 $k$는 1이상이어야 한다. 만일 $k$가 정수이면, 최초 원본 데이터에서 분할된 $m$개의 조각들은 모두 $k$만큼 복제되어 각 스토리지 서버에 저장된다. 만일 $k$가 소수이면, 원본 데이터의 일부 조각은 $\lfloor k \rfloor + 1$만큼 복제되고 다른 조각들은 $\lfloor k \rfloor$ 만큼 복제되어 각 스토리지 서버에 저장된다. (그림 1)은 (3, 8)-IDS의 예를 나타내고 있다. 그림에서 나타나듯, 조각 1, 2는 3배 만큼 복제 되었고, 반면에 조각 3는 2배 만큼 복제 되었다.

**[정리 1]** $(m, n)$-IDS를 사용하는 시스템이 묵인할 수 있는 임의 서버의 최대 고장 수는 다음과 같다.

$$\lfloor k \rfloor - 1$$

**증명** : 일반적으로 $(m, n)$-IDS를 사용하는 시스템은 $n$-$m$개의 스토리지 서버의 고장을 묵인할 수 있다. 그러나 이 서버의 고장 수는 원본 데이터를 재구성할 수 있는 $m$개의 데이터 조각을 저장하고 있는 서버들은 고장나지 않는다는 가정이 내포되어 있기 때문에 임의적이지 못하다. 위에서 설명했듯이, $k$가 정수라고 가정하면 $(m, n)$-IDS을 사용하는 시스템은 최초 원본 데이터의 $m$개의 데이터 조각들을 $k$만큼 복제하여 각 스토리지 서버에 분산 저장한다. 그러므로 시스템 사용자가 원본 데이터를 액세스하려 할 때 데이터가 가용하기 위해서는 $k$개의 같은 조각을 저장하고 있는 서버들 중 최소한 하나만이라도 가용해야만 한다. 따라서 $(m, n)$-IDS를 사용하는 시스템이 묵인할 수 있는 임의 서버의 최대 고장 수는 $k$-1이다. 만일 $k$가 소수이면 시스템이 묵인할 수 있는 임의 서버의 고장 수는 $\lfloor k \rfloor - 1$이다. ■

(그림 1)에서 두 개의 데이터 조각 3을 저장하고 있는 서버가 고장나면 나머지 6개의 조각들을 저장하고 있는 서버들로는 원본 데이터를 재구성할 수 없다. 따라서 (3, 8)-IDS가 묵인할 수 있는 임의 서버의 최대 고장 수는 [정리 1]에 의해 1이다. 일반적으로 IDS를 사용하는 서바이벌 스토리지 시스템들은 데이터 조각들을 일정하게 복제한다. 따라서 본 논문에서 언급하는 모든 IDS들은 다음과 같다고 가정한다.

**[가정 1]** 서바이벌 스토리지 시스템에서 사용되는 모든 IDS들은 정수인 IER를 갖는다.

### 2.3 IDS의 가용성 공식

예전의 논문들은 IDS의 가용성을 평가하기 위해 다음과 같은 고전적인 가용성 공식을 이용하였다.

$$(m, n)\text{-IDS의 가용성} = \sum_{i=m}^{n} \binom{n}{i} P_s^i (1-P_s)^{n-i}$$

그러나 이 가용성 공식은 정확하지 못하다. 왜냐하면 데이터 조각의 종류에 상관없이 $m$개의 데이터 조각을 저장하고 있는 서버가 가용하면 원본 데이터가 가용한 걸로 계산하기 때문이다. 예를 들어 (그림 1)에서 두 개의 데이터 조각 3를 저장하고 있는 서버가 고장 날 경우 원본 데이터는 가용하지 못하다. 그러나 위 고전적인 가용성 공식은 이러한 경우도 가용한 걸로 포함을 시킨다. 다시 설명하면 어떠한 경우라도 원본 데이터를 재구성하는데 필요한 데이터 조각 개수 이상으로 데이터 조각들이 가용하면 데이터가 가용한 걸로 계산한다. 따라서 본 논문에서 좀 더 정확한 가용성 공식을 제안한다. 그 공식은 다음과 같다.

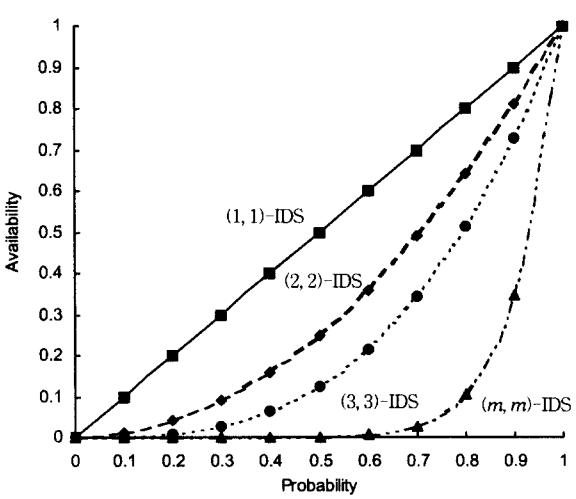
$$(m, n)\text{-IDS의 가용성} = \left( \sum_{i=1}^{k} \binom{k}{i} P_s^i (1-P_s)^{k-i} \right)^m$$

본 논문의 나머지 부분에서는 다음 가정하에 이 새로운 가용성 공식을 이용하여 IDS의 기본적인 특성에 대해서 설명한다.

**[가정 2]** 모든 스토리지 서버들은 동일한 성공확률을 가지며, 고장은 모두 독립적이다[5].

### 2.4 IDS의 선택의 어려움

$(i, i)$-IDS의 가용성은 $P_s$가 고정된 상태에서 $i$가 증가할수록 감소한다. (그림 2)는 $Class_1$의 IDS의 가용성을 나타내고 있다. 그림에 나타나듯 IER이 1인 상태에서 데이터를 분할 할수록 가용성은 개선되지 않는다는 것을 알 수 있다[6].

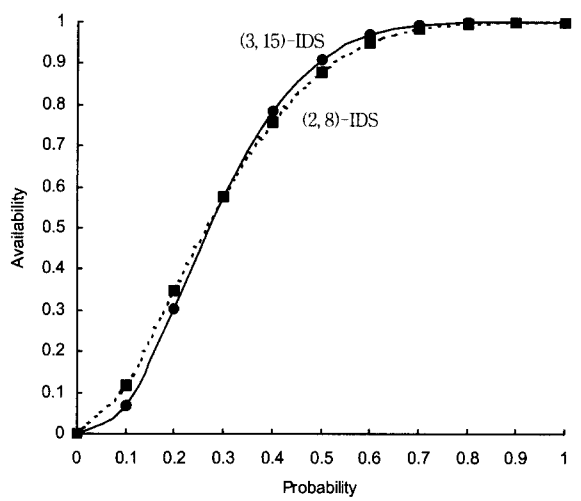



(그림 2) $Class_1$의 IDS들의 가용성

$Class_4$의 (2, 8)-IDS와 $Class_5$의 (3, 15)-IDS의 가용성은 다음과 같은 관계를 갖는다.

$$P_d(2,8) > P_d(3,15),\ P_s < 0.3$$
$$P_d(2,8) = P_d(3,15),\ P_s = 0.3$$
$$P_d(2,8) < P_d(3,15),\ P_s > 0.3$$

다시 말해서, $P_s < 0.3$이면 (2, 8)-IDS의 가용성이 (3, 15)-IDS보다 크다. 반면, $P_s > 0.3$이면 (3, 15)-IDS의 가용성이 (2, 8)-IDS보다 크다. (그림 3)은 이들의 가용성을 나타내고 있다[6]. 이 예에서 나타나듯, 하나 이상의 IDS가 있을 때 $P_s$의 구간 마다 최적인 IDS가 다르다는 것을 알 수 있다. 달리 말하면, 큰 IER를 갖는 IDS가 적은 IER를 갖는 IDS보다 적은 가용성을 가질 수도 있다. 이처럼 IER만으로는 최적인 IDS를 구별하기가 힘들다.




(그림 3) (2, 8)-IDS와 (3, 15)-IDS의 가용성

## 3. 최적인 IDS

이번 장에서는 최적인 IDS를 찾는데 도움이 되는 IDS들의 특성들에 대해 알아본다.

**[정리 2]**

$$P_d(m,n) < P_d(m, n+mi),\ i > 1$$

**증명** : $P_s$의 모든 범위에서 $(m, n+mi)$-IDS의 가용성은$(m, n)$-IDS보다 크다. 다시 말해서, $m$이 고정된 상태에서 $i$가 증가할수록 $(m, n+mi)$-IDS의 가용성은 증가한다. 임계 확률은 $(m, n+mi)$-IDS와 $(m, n)$-IDS의 사이에 존재하지 않는다. [정리 2]는 시스템 설계자에게 데이터를 분산 저장할 때 시스템의 각 서버에 데이터가 저장되도록 IDS를 설계하라는 것을 알려 주고 있다. $(m, n+mi)$-IDS와 $(m, n)$-IDS는 원본 데이터를 재구성할 때 $m$개의 조각이 필요하다. 그리고 $(m, n)$-IDS를 사용하는 시스템이 묵인할 수 있는 임의 서버의 최대 고장 수는 [정리 1]에 의해 $k$-1이다. $(m, n+mi)$-IDS를 사용하는 시스템은 $k+i$-1이다. 따라서 $(m, n+mi)$-IDS는 $i$가 증가할수록 가용성이 증가하고, 시스템이 묵인할 수 있는 임의 서버의 고장 수가 증가 한다는 것을 알 수 있다. ■

$(m, n+mi)$-IDS를 사용하는 시스템에 새로운 서버를 추가하여도 각 서버에 저장되어 있는 데이터 조각들을 변경하거나 업데이트 할 필요는 없다. 다만 서버의 수를 축소할 때는 상황은 달라진다. $i$값이 큰 IDS를 이용할 때는 IER를 $k$에서 $k+i$로 증가시키는 비용이 필요하다. IER이 증가한다는 것은 스토리지 비용이 증가한다는 것을 뜻한다[6].

**[정리 3]**

$$P_d(m,n) > P_d(m+i,n),\ n-m \geq i > 0 \text{이고} \frac{n}{m+i} \text{는 정수}$$

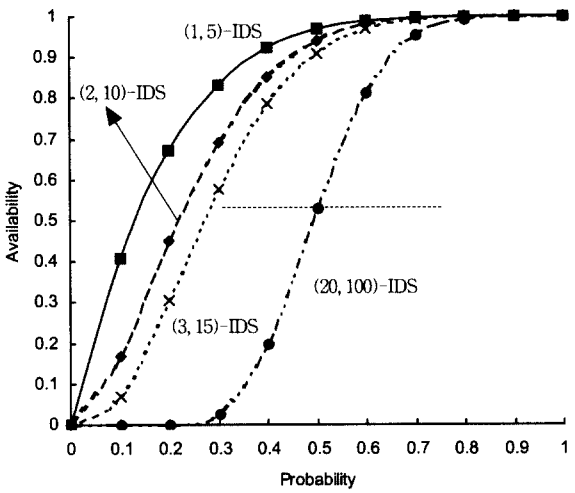
**증명** : $(m+i, n)$-IDS에서 $i$가 $n-m$보다 크면 원본 데이터는 손실이 발생한다. 따라서 $i$는 $n-m$보다 작아야 한다. [정리 1]에 의해 $(m, n)$-IDS은 $k$-1개의 노드 고장을 묵인할 수 있다. 반면 $(m+i, n)$-IDS는 그보다 적은 $\frac{n}{m+i}-1$개의 노드 고장을 묵인할 수 있다. 따라서 $(m, n)$-IDS이 $(m+i, n)$-IDS보다 더 가용하다. ■

[정리 3]에서 $(m, n)$-IDS의 가용성은 $P_s$의 모든 범위에서 $(m+i, n)$-IDS보다 크다. 다시 말해서 $(m, n)$-IDS와 $(m+i, n)$-IDS는 임계 확률을 갖지 않는다. [정리 3]은 데이터가 저장되는 서버의 수가 고정된 상태에서 데이터를 분할할수록 가용성이 감소한다는 것을 뜻한다. 그리고 시스템의 서버수가 고정 되어 있다면 각 서버에 가능한 한 큰 데이터 조각을 저장해야 한다는 것을 알려주고 있다. $(m+i, n)$-IDS는 $i$가 작을수록 IER이 증가하기 때문에 좋은 가용성을 갖는다[6].

**[정리 4]**

$$P_d(m,n) > P_s(m+i,n+j),\ k=\frac{n}{m}=\frac{n+j}{m+i},\ i,j \geq 1$$

**증명** : 가용성 공식 $P=(f(k, p))^m$에서 $f(k, p) < 1$이고 $k$, $p$가 고정되어 있을 때, $m$이 증가 할수록 $P$가 감소한다는 것을 알 수 있다. ■




(그림 4) $Class_5$의 IDS들의 가용성

[정리 4]에서 $(m, n)$-IDS와 $(m+i, n+j)$-IDS를 이용하는 시스템이 묵인할 수 있는 임의 서버의 최대 고장 수는 같지만, 주의할 것은 임의의 한 $Class_k$의 모든 IDS들이 서로 임계 확률을 갖지 않는다는 것이다. 그리고 [정리 4]에 의해 모든 IDS들이 같은 IER을 갖지만 $P_s$의 모든 범위에서 가장 최적인 IDS는 $(1, k)$-IDS이다. IER이 고정된 상태에서 $(m+i, n+j)$-IDS의 $i$와 $j$가 증가할수록 가용성이 감소한다. 다시 말해서 [정리 4]는 IER이 고정된 상태에서 분할 수와 서버에 저장되는 수가 증가하여도 가용성에는 전혀 도움이 안된다는 것을 나타내고 있다.

(그림 4)는 $Class_5$의 IDS들의 가용성을 나타내고 있다. 여기서 우리는 [정리 4]를 통해 각 클래스(Class)의 IDS들 중 (1, 1), (1, 2), (1, 3), …, (1, n)-IDS들이 각 클래스에서 최적이라는 것을 알 수 있다.

**[정리 5]**

$$P_d(i,j) < P_d(m,n),\ i < m, j < n$$

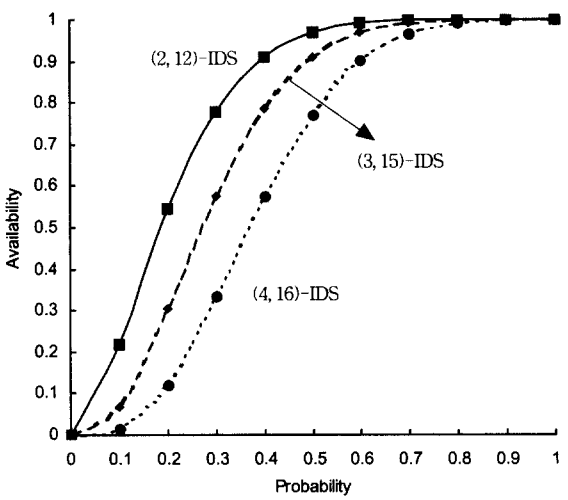
**증명** : [정리 2]에 의해 $j < n$일 때, $P_d(i,j) < P_d(i,n)$. [정리 3]에 의해 $i > m$일 때, $P_d(i,n) < P_d(m,n)$. 따라서 $i > m$과 $j < n$일 때, $P_d(i,j) < P_d(m,n)$이다. ■

[정리 5]에서 $(m, n)$-IDS의 가용성은 $P_s$의 모든 범위에서 $(i, j)$-IDS보다 크다. $i$보다 적은 개수의 데이터 조각을 $j$보다 많은 서버에 저장하기 때문이다. $(m, n)$-IDS와 $(i, j)$-IDS는 임계 확률을 갖지 않는다. [정리 5]는 데이터 조각을 분산 저장할 때, 큰 데이터 조각으로 가능한 한 많은 서버에 저장해야 한다는 것을 나타내고 있다. 이 방법은 원본 데이터를 재구성할 때 적은 데이터 조각이 필요하므로 가용성이 증가한다. 이러한 장점은 IER의 큰 비용으로 달성된다[6].

**[정리 6]**

$$P_d(i,j) > P_d(m,n),\ i < m, j < n \text{ and } \frac{j}{i} \geq \frac{n}{m}$$

**증명** : $(m, n)$-IDS와 $(i, j)$-IDS의 IER이 같은 경우에는 [정리 4]에 의해 $(i, j)$-IDS의 가용성이 $(m, n)$-IDS보다 크다는 것을 알 수 있다. $(i, j)$-IDS의 IER이 $(m, n)$-IDS보다 큰 경우는 뒤에서 설명하게 될 [정리 9]에 의해 $(i, j)$-IDS가 더 가용 하다는 것을 알 수 있다. 그러므로 $P_s$의 모든 범위에서 $(i, j)$-IDS의 가용성이 $(m, n)$-IDS보다 크다. ■




(그림 5) (2, 12)-IDS, (3, 15)-IDS, (4, 16)-IDS의 가용성

[정리 5]처럼 [정리 6]도 분할되는 개수가 작고 서버에 분산 저장되는 데이터 조각이 많을수록 가용성이 증가한다는 것을 나타내고 있다. (그림 5)는 [정리 6]의 예를 나타내고 있다.

**[정리 7]**

$$P_d(k_1 m, k_1 n) > P_d(k_2 m, k_2 n),\ k_1 < k_2$$

**증명** : $(k_1 m, k_1 n)$-IDS와 $(k_1 m, k_1 n)$-IDS는 IER이 $n/m$ 인 같은 클래스에 속한다. 그러므로 [정리 4]에 의해 $(k_1 m, k_1 n)$-IDS의 가용성이 $(k_1 m, k_1 n)$-IDS보다 더 크다는 것을 알 수 있다. 그리고 두 IDS간 임계 확률은 존재하지 않는다. ■

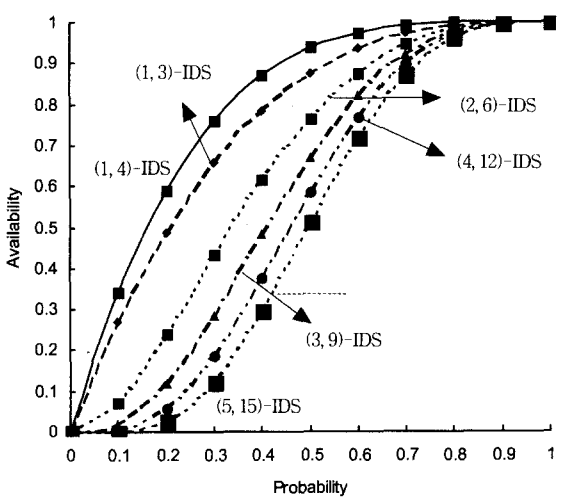
**[정리 8]** $(i, j)$-IDS와 $(k, l)$-IDS 사이에 임계 확률이 존재하지 않고 $(k, l)$-IDS의 IER이 $(i, j)$-IDS보다 큰 경우 의 모든 범위에서 $(k, l)$-IDS의 가용성이 더 크다. $(i, j)$-IDS와 $(k, l)$-IDS 사이에 임계 확률이 존재하며, $(k, l)$-IDS의 IER이 $(i, j)$-IDS보다 큰 경우에는 다음과 관계를 갖는다.

$$P_d(i,j) > P_d(k,l), \quad 0 < P_s < P_s^*((i,j),(k,l))$$
$$P_d(i,j) > P_d(k,l), \quad P_s = P_s^*(i,j),(k,l)$$
$$P_d(i,j) > P_d(k,l), \quad P_s^*(i,j),(k,l) < P_s < 1$$

[정리 8]은 두 IDS 중 한 IDS가 $P_s$의 일정 범위 내에서 다른 IDS보다 가용성이 크고 다른 범위에서는 작다는 것을 나타내고 있다. 이처럼 조건에 따라 임의의 한 IDS가 다른 IDS들보다 항상 가용성이 큰 것은 아니다. [정리 8]은 IDS를 선택할 때, 의 범위를 먼저 선택한 다음 적당한 IDS를 선택하라는 것을 제시하고 있다[6].

**[정리 9]** IER이 다른 $Class_a$와 $Class_b$가 주어졌을 때, 두 클래스의 IER의 관계가 이라면, $Class_a$의 임의의 $(i, j)$-IDS는 $Class_b$에 속하는 IDS들에 대해 바운드 IDS인 -IDS를 갖는다. 이 경우 $m, n$은 $i, j$보다 크다. 반면 두 클래스의 IER의 관계가 이라면, 바운드 IDS는 $i, j$보다 큰 범위에서는 존재하지 않고, 작은 범위에 존재하거나 존재하지 않을수도 있다.

**증명** : $a < b$인 경우에서 $m$과 $n$이 $i$와 $j$보다 작을 경우, [정리 6]에 의해 $(m, n)$-ID이 $(i, j)$-IDS보다 더 가용하다. 다시 말해 임계확률은 존재하지 않는다. 그런데 $l \to \infty$이면 [정리 4]에 의해 $p(l, bl) \to 0$이다. 그러므로 바운드 IDS, $(m, n)_{(i,j)}$-IDS는 $i$와 $j$보다 큰 영역에 반드시 존재한다. 마찬가지로 $a > b$인 경우에서 $m$과 $n$이 $i$와 $j$보다 클 경우, [정리 6]에 의해 $(i, j)$-IDS이 $(m, n)$-ID보다 더 가용하다. 만일 $l \to 0$이면 [정리 4]에 의해 $p(l, bl) \to 0$이다. 그러므로 $(m, n)_{(i,j)}$-IDS는 $i$와 $j$보다 작은 영역에 반드시 존재한다. ■



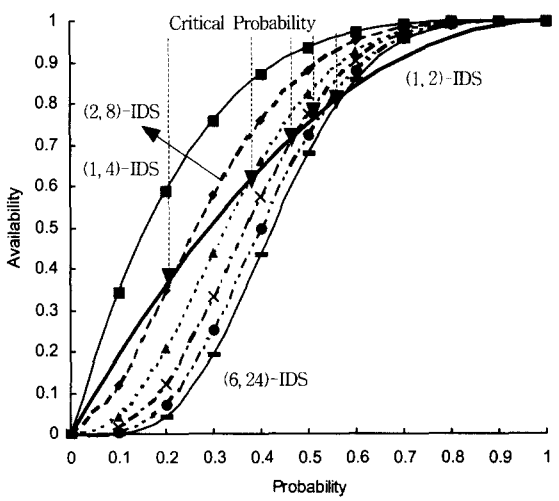

(그림 6) (1, 4)-IDS와 $Class_3$의 IDS들의 가용성

우리는 [정리 8]에 의해 $(m, n)$-IDS와 $Class_a$가 바운드 IDS를 갖지 않고 $(m, n)$-IDS의 IER이 $Class_a$보다 큰 경우 $(m, n)$-IDS의 가용성은 $P_s$의 모든 범위에서 $Class_a$의 모든 IDS들보다 더 크다는 것을 알 수 있다. (그림 6)은 [정리 9]의 예를 나타내고 있다.

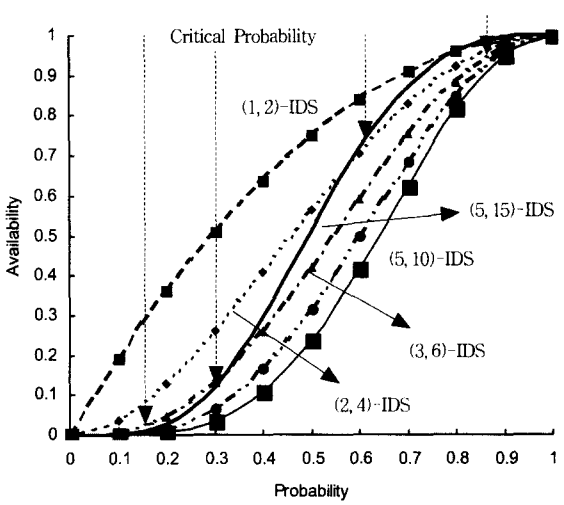
일반적으로 (1, 1), (1, 2), (1, 3), …, (1, $n$)-IDS들은 자신보다 IER이 작은 클래스들과는 바운드 IDS를 갖지 않는다.

**[정리 10]** $Class_a$의 IER이 $Class_b$보다 크고 $Class_a$의 $(i, j)$-IDS가 $Class_b$와 -IDS를 갖는다고 가정하자. 그러면 $m, n$이 증가할수록, 임계 확률은 1에 가까워진다. 반대로 $(m, n)_{(i,j)}$-IDS은 갖는데 $Class_b$의 IER이 크다면, $m, n$이 감소할수록 임계 확률이 1에 가까워진다.

**증명** : $Class_b$에서 가장 가용한 IDS는 [정리 4]에 의해 (1, $b$)-IDS이다. 그리고 $l \to \infty$이면 $p(l, bl) \to 0$이다. 그러므로 $a < b$ 경우에 $m$과 $n$이 증가할 때, $Class_a$의 $(i, j)$-IDS와 $(m, n)_{(i,j)}$-IDS의 임계 확률은 1에 가까워 진다. $a > b$ 경우에는 0에 가까워진다. ■




(그림 7) (1, 2)-IDS와 $Class_4$의 가용성




(그림 8) (5, 15)-IDS와 $Class_2$의 가용성

(그림 7)과 (그림 8)은 [정리 10]의 예를 각각 나타내고 있다. (그림 7)에서 (1, 2)-IDS의 바운드 IDS는 (2, 8)-IDS이다. 그리고 (1, 2)-IDS와 (1, 4)-IDS는 임계 확률을 갖지 않는다는 것을 알 수 있다. (그림 8)에서는 (5, 15)-IDS의 바운드 IDS는 (4, 8)-IDS이다. 마찬가지로 (5, 10)-IDS와 (5, 15)-IDS는 임계 확률을 갖지 않는다는 것을 알 수 있다.

## 4. 최적인 IDS

지금까지 정보 분할 기법의 특성에 대해서 알아보았다. 시스템 설계자들은 가능한 IDS 집합이 주어졌을 때 최적인 IDS를 찾고자 할 것이다. 이번 장에서는 IDS 집합이 주어졌을 때 3장에서 알아본 IDS의 특성들을 이용하여 가용성 관점에서 최적인 IDS를 찾는 알고리즘을 제안한다.

IDS 집합은 3장의 정리들을 이용하여 줄일 수 있다. 알고리즘의 입력은 시스템 설계자가 여러 가지 제약들을 고려하여 선택할 수 있는 IDS들의 집합이다. 알고리즘의 결과는 가능한 최적인 IDS만 남아있는 IDS수가 줄어든 집합이다.

- 입력 S : 시스템 설계자가 선택할 수 있는 IDS들의 집합
- 출력 S′ : IDS수가 줄어든 집합

단계 1 : 입력 S′ = S
단계 2 : S′을 클래스 별로 분류한다.
단계 3 : 클래스 마다 $m$, $n$이 가장 작은 $(m, n)$-IDS를 선택한다. 나머지는 삭제한다.
단계 4 : IER이 가장 큰 클래스의 $(m, n)$-IDS를 다른 클래스의 IDS과 분할 개수를 비교하여 같거나 작으면 $(m, n)$-IDS를 선택하고 나머지는 삭제한다.
단계 5 : IDS들이 단계 4를 만족하지 않으면, IER이 가장 큰 클래스의 $(m, n)$-IDS와 임계 확률을 갖는 IDS들을 선택한다. 나머지는 삭제한다.
단계 6 : 출력 S′

(Algorithm 1)

단계 3은 [정리 4]를 기초로 하고 있다. 같은 클래스의 IDS들은 서로 임계 확률을 갖지 않기 때문에 클래스에서 가장 최적인 IDS는 $m$, $n$이 가장 작은 $(m, n)$-IDS이다. 그러므로 클래스에서 가장 최적인 IDS 개수는 1이다. 만일 단계 3에서 $g$개의 클래스가 있다면, 단계 3의 결과로 남는 IDS의 개수는 $g$개이다. 단계 4는 [정리 2]와 [정리 5]를 기초로 하고 있다. 만일 $g$개의 IDS들이 단계 4를 만족하면, 최초 입력인 IDS 집합으로부터 가장 최적인 IDS 개수는 1이다. 그렇지 않으면 하나 이상이다. 그러면 의 범위마다 다른 최적인 IDS가 존재한다. (그림 3)은 두 개의 IDS의 경우를 보이고 있다. $0 < P_s < 0.3$이면, 가장 최적인 IDS는 (2, 8)-IDS이고, $0.3 < P_s < 1$에서는 (3, 15)-IDS가 최적이다.

## 5. 결론 및 향후 계획

지금까지 이 논문을 통해 정보 분할 기법의 특성에 대해서 알아보았다. 그리고 그러한 특성들을 이용하여 최적인 IDS를 결정하는 알고리즘을 개발하였다. 정보 분할 기법의 중요한 특성은 가능한 한 큰 데이터 조각을 모든 서버에 하나씩 저장시켜야 한다는 것과 시스템 설계자는 $P_s$의 범위를 선정한 다음 적당한 IDS를 선정해야 한다는 것이다. 이 논문에서 IDS의 가용성의 특성들을 파악할 때 몇 가지 가정들을 했다. 그러므로 현실환경에는 거리가 있다. 앞으로 계획으로 좀더 현실세계에 가까운 모델로 발전시킬 것이다.

## 참 고 문 헌


[1] J. J. Wylie, M. W. Bigrigg, J. D. Strunk, G. R. Ganger, H. Kiliccote, P. K. Khosla, "Survivable information storage systems," IEEE Computer, pp.61-68, 2000.

[2] M. O. Rabin, "Efficient Dispersal of Information for Security, Load Balancing and Fault Tolerance," ACM, pp.335-348, 1989.

[3] A. Shamir, "How to Share a Secret," Comm. ACM, Vol.22, pp.612-613, 1979.

[4] G. R. Blakley, C. Meadows, "Security of Ramp Schemes," Advances in Cryptology CRYPTO, pp.242-268, 1985.

[5] J. J. Wylie, M. Bakkaloglu, V. Pandurangan, M. W. Bigrigg, S. Oguz, K. Tew, C. Williams, G. R. Ganger, P. K. Khosla, "Selecting the Right Data Distribution Scheme for a Survivable Storage System," Technical Report CMU-CS-01-120 Carnegie Mellon University, 2001.

[6] Hung-Min Sun, Shiuh-Pyng Shieh, "Optimal Information Dispersal for Increasing the Reliability of a Distributed Service," IEEE Trans. Vol.46, pp.462-472, 1997.



**송 성 근**

e-mail : kkskk103@skku.edu
2000년 성균관대학교 전기전자컴퓨터 공학부 학사
2002년~현재 성균관대학교 전기전자 컴퓨터공학부 석사과정
관심분야 : 정보보안, 모바일 컴퓨팅, 시스템 소프트웨어 등

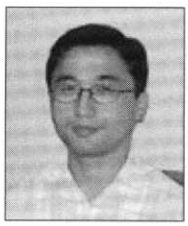


**윤 희 용**

e-mail : youn@ece.skku.ac.kr
1977년 서울대 전자공학과 학사
1979년 서울대 전자공학과 석사
1988년 Univ. of Massachusetts 컴퓨터공학과 박사
1988년~1991년Univ. of North Texas, 조교수
1991년~1999년 Univ. of Texas at Arlington 부교수
1999년~2000년 정보통신대학교 교수
2000년~현재 성균관대학교 정보통신공학부 교수
관심분야 : 모바일 컴퓨팅, 분산처리, 시스템 소프트웨어

이 형 수

e-mail : hslee@keti.re.kr
1989년 한양대학교 전자공학과 학사
1989년~1997년 LG전자 미디어통신연구소
2000년 아주대학교 컴퓨터공학과 석사
1997년~현재 전자부품연구원 정보시스템 연구센터 책임연구원
관심분야 : 디지털 통신 프로토콜, 네트워크 저장 장치, 내장형 시스템



이 강 신

e-mail : kslee@kisa.or.kr
1987년 한양대학교 수학과(석사)
1989년 한양대학교 수리통계학과(석사)
1990년~1992년 (주)데이콤종합연구소 연구원
1992년~2000년 한국전산원 부장
2000년~현재 한국정보보호진흥원 기반보호기술팀 팀장
관심분야 : 정보보안, 침입감내, 무선보안 등