리눅스 클러스터 화일 시스템 SANique™의 오류 회복 기법

이 규 웅†

요 약

본 논문은 SAN(storage area network)상에 네트워크-부착형(network-attached) 저장 장치들을 직접 연결하여 화일 서버 없이 직접 데이타 전송이 가능한 SAN 기반의 리눅스 클러스터 공유 화일 시스템인 SANique™에 대한 설계방법을 기술하며, 각 시스템 구성 요소인 CFM(Cluster File Manager), CVM(Cluster Volume Manager), CLM(Cluster Lock Manager), CBM(Cluster Buffer Manager), CRM(Cluster Recovery Manager)에 대한 특징들을 설명한다. 또한 클러스터 내의 노드 오류에 의해 발생하는 "split-brain" 오류 상황 및 문제점을 공유 화일 시스템 환경 하에서 정의하고, SAN 디스크인 공유 디스크 보드를 활용하여 "split-brain" 문제를 해결하는 방법과 공유 화일 시스템의 오류 회복 알고리즘을 제시한다.

Failure Recovery in the Linux Cluster File System SANique™

Kyu Woong Lee†

ABSTRACT

This paper overviews the design of SANique™ - a shred file system for Linux cluster based on SAN environment. SANique™ has the capability of transferring user data from network-attached SAN disks to client applications directly without the control of centralized file server system. The paper also presents the characteristics of each SANique™ subsystem : CFM(Cluster File Manager), CVM(Cluster Volume Manager), CLM(Cluster Lock Manager), CBM(Cluster Buffer Manager) and CRM(Cluster Recovery Manager). Under the SANique™ design layout, then, the syndrome of "split-brain" in shared file system environments is described and defined. The work first generalizes and illustrates possible situations in each of which a shared file system environment may split into two or more pieces of separate brain. Finally, the work describes the SANique™ approach to the given "split-brain" problem using SAN disk named "shared disk board" and develops the overall recovery procedure of shared file systems.

키워드 : SAN(Storage Area Network), 클러스터 시스템(Cluster System), Split-brain 오류 (Failure), 회복(Recovery), 화일 시스템 (File System)

1. 서 론

네트워크의 발전으로 인하여 지역적 화일 시스템(local file system)은 점차 서버들의 클러스터화를 통한 분산 화일 시스템(distributed file system) 또는 공유 화일 시스템(shared file system)으로 발전되고 있다. 전형적 클라이언트-서버 분산 화일 시스템은 중앙 집중적 서버에 의해 화일 식별 공간, 화일 접근 허가 권한 등을 제공받으며, 화일 이름과 화일 오프셋을 디스크 블록 어드레스로 매핑하는 기능을 제공받고 있다. 이와 같은 클라이언트-서버 분산 화일 시스템은 각 클라이언트에서 제공되는 지역 화일 시스템 명령어에 의해 분산 저장된 화일을 접근할 수 있으며 또한 RPC, XDR, TCP/IP 등과 같은 표준 네트워크 프로토콜에 의해 각 클라이언트 시스템의 네트워크 하드웨어와 프로토콜에 대한 독립성을 제공받을 수 있다. 전형적인 클라이언트-서버 분산 화일 시스템은 클라이언트의 요청을 처리하기 위한 중앙 집중적인 하나의 서버를 갖는다. 이와 같은 시스템은 서버 노드의 오류에 의한 제한된 확장성과 가용성을 제공한다는 성능상의 단점을 갖게 된다. 이러한 분산 화일 시스템의 대표적인 예는 썬 마이크로 시스템사의 NFS(Network File System)과 AT&T사의 RFS(Remote File Sharing System), IBM과 CMU 대학에서 초기 개발한 AFS(Andrew File System) 등이 있다[5, 6, 12].

최근 화이버 채널 인터페이스 기술 발전으로 인한 네트워크-부착형(network-attached) 저장 장치들이 등장함에 따라, 네트워크 프로토콜 스택을 갖춰야 하는 NFS와 같은 전형적인 분산 화일 시스템의 구조가 SAN 기반의 공유 화일

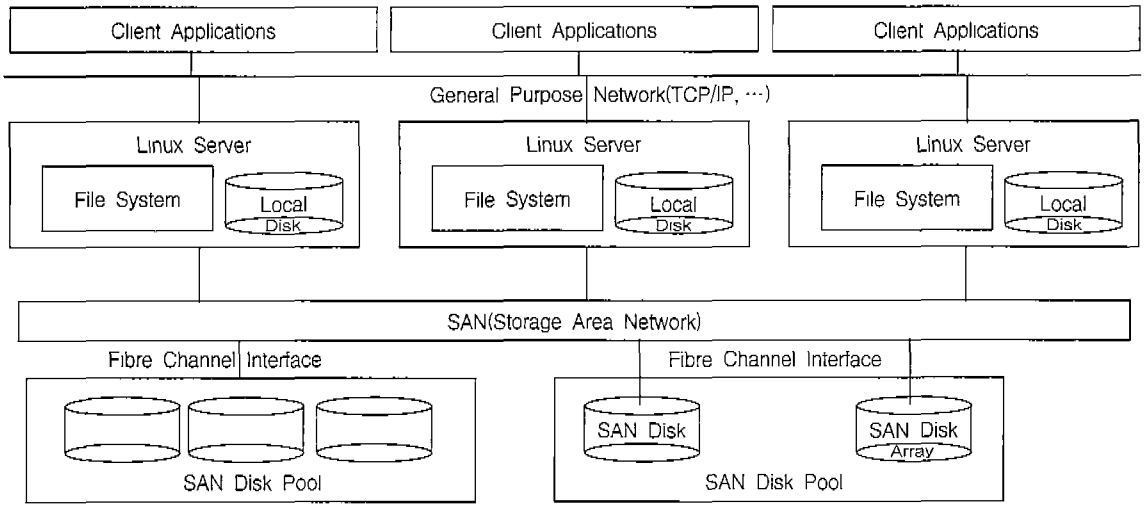
※ 이 논문은 (주)네크로 임직원 연구비 지원에 의한 것임.
† 정 회 원 : 상지대학교 컴퓨터정보공학부 교수
논문접수 : 2001년 10월 4일, 심사완료 : 2001년 12월 24일

시스템 구조로 변화되고 있다. 참고문헌 [2]에서 공유 화일 시스템은 "클라이언트와 다수의 클라이언트가 같은 저장 장치로부터 동시 접근할 수 있는 저장장치 사이에 데이타를 "직접적"으로 전송할 수 있는 화일 시스템"이라 정의하고 있다. SAN 기반 공유 화일 시스템은 (그림 1)과 같이 분산 화일 시스템의 기능을 모두 제공하며, 또한 네트워크 부착형 저장 장치를 서버 없이 직업 저장 장치 전용 네트워크(SAN)에 접속시켜 사용하므로 가용성 및 확장성에 있어서 기존 분산 화일 시스템 보다 우수하다. SAN 공유 화일 시스템은 기존 분산 시스템에서 서버가 모든 화일 공유의 제어를 담당해야 하는 단점을 극복할 수 있는 새로운 구조의 분산 화일 시스템이다.

공유 화일 시스템은 설계 방법에 따라 대칭형(symmetric)과 비대칭형(asymmetric)으로 분류된다[2]. 대칭형 설계 방법은 (그림 1)에서 하나의 리눅스 서버가 화일 시스템 연산(읽기, 쓰기)을 수행하기 위하여 어떠한 리눅스 서버와도 통신 없이 수행할 수 있는 공유 화일 시스템을 의미한다. 즉, 한 서버가 공유 화일 시스템을 접근하기 위한 모든 능력을 제공한다. 반면에, 비대칭 공유 화일 시스템은 공유 화일 시스템에 대한 화일 관리기 기능을 맡고 있는 임의의 서버가 존재하는 경우로서, 하나의 서버에서 발생하는 공유 화일 접근 연산은 반드시 화일 관리기능을 담당하고 있는 서버에 의존해야만 데이타 접근이 가능하게 된다. 비대칭형 공유 화일 시스템은 공유 화일 관리를 위한 메타 데이타(meta data)의 유지 및 갱신을 책임지고 있는 화일 관리기 역할의 서버가 존재하므로, 이 서버로 인한 데이타 요청 처리에 대한 병목 현상이 발생가능하며, 그 서버의 오류로 인한 전체 공유 화일 시스템의 오류가 발생할 수 있는 단점을 가지고 있다. 반면에 대칭형 공유 화일 시스템은 모든 서버들이 화일 관리기 기능을 유지해야 하므로 공유 화일 시스템 유지를 위한 메타 데이타의 중복 저장이 불가피하며, 이로 인한 메타 데이타의 불일치성(inconsistency)이 발생할 수 있으며, 메타 데이타 일치성을 보장하기 위해 비대칭형 공유 화일 시스템보다 더 많은 데이타 통신이 필요하게 된다. 대표적인 공유 화일 시스템의 예로서는 Veritas사의 CFS, VxFS, California 대학(UCB)의 xFS, Minnesota 대학의 Global File Systems, Oracle사의 Parallel Database Server 등이 있다[7-11]

SAN 기반 공유 화일 시스템은 중앙 집중적인 화일 서버의 참여 없이(serverless) 저장 장치로부터 직접 데이타를 접근해야 하므로 기존 분산 시스템에서 적용하던 화일 관리 기법, 즉, 메타 데이타 일치성 보장 및 오류 회복 기법을 재설계해야 한다. SANique^{TM}은 리눅스 클러스터 시스템을 기반의 SAN 기반 공유 화일 시스템으로서 I/O 병목현상을 제거하고 확장성 및 가용성을 극대화 하였으며, 특정 노드 및 디스크 오류에도 온라인 상에서 회복이 가능한 공유 화일 시스템이다[1]. 본 논문에서는 SANique^{TM}의 기본 시스템 구조 및 SANique^{TM} CFM(Cluster File Manager), CLM(Cluster Lock Manager), CVM(Cluster Volume Manager), CBM(Cluster Buffer Manager), CRM(Cluster Recovery Manager)의 기능을 설명한다. 또한 공유 화일 시스템상에서 발생 가능한 오류의 종류 및 문제점을 기술하고 온라인 상태에서 회복 가능한 오류 회복 기법을 기술한다. 특히 오류 발생시 공유 화일 시스템에서 발생하는 "split-brain" 문제를 설명하고 SAN 디스크의 활용을 통한 문제 해결 방안을 제시한다.

본 논문의 구성은 다음과 같다. 제 2장에서는 SANique^{TM}의 기본 시스템 구성도 및 각 모듈별 기능을 설명한다. 제 3장에서는 오류의 종류 및 탐지기법, 문제점을 설명하고, 특히 전형적인 분산 시스템에서 발생하는 "split-brain" 문제를 새로운 화일 시스템 구조인 공유 화일 시스템 구조상에서 정의하고 이에 대한 문제점들을 조사한다. 공유 디스크 보드라는 SAN 디스크의 활용을 통해 오류가 발생한 노드를 정확히 파악하고 그 노드를 공유 화일 시스템 그룹에서 제거하여 "split-brain" 문제를 해결하는 방법을 제시하고, 이 방법을 활용한 공유 화일 시스템의 오류 회복 기법을 제 4장에서 설명한다. 끝으로 제 5장에서 본 논문의 결론과 향후 연구방향을 기술한다.



(그림 1) SAN 기반의 공유 화일 시스템 구성도

2. SANique™의 시스템 구성도

SANique™은 리눅스 클러스터 서버를 위한 SAN 기반의 저장 장치 관리 솔루션이다. SANique™은 전형적인 분산 화일 서버 시스템 구조의 병목 현상을 제거하고 I/O 대역폭을 늘려 성능을 최대화한다. 또한 저장 장치로부터 응용에까지 화일 서버의 경유 없이 직접적인 데이타 전송을 하므로 데이타 전송 최단 경로를 보장하며 병렬 데이타 접근을 보장하여 응용별 I/O 대역폭을 증가시킬 수 있다. SANique™은 클러스터 화일 관리기(CFM), 클러스터 볼륨 관리기(CVM), 클러스터 로크 관리기(CLM), 클러스터 버퍼 관리기(CBM), 클러스터 회복 관리기(CRM)로 구성되며, 이 서브 시스템 간의 구성은 (그림 2)와 같다.

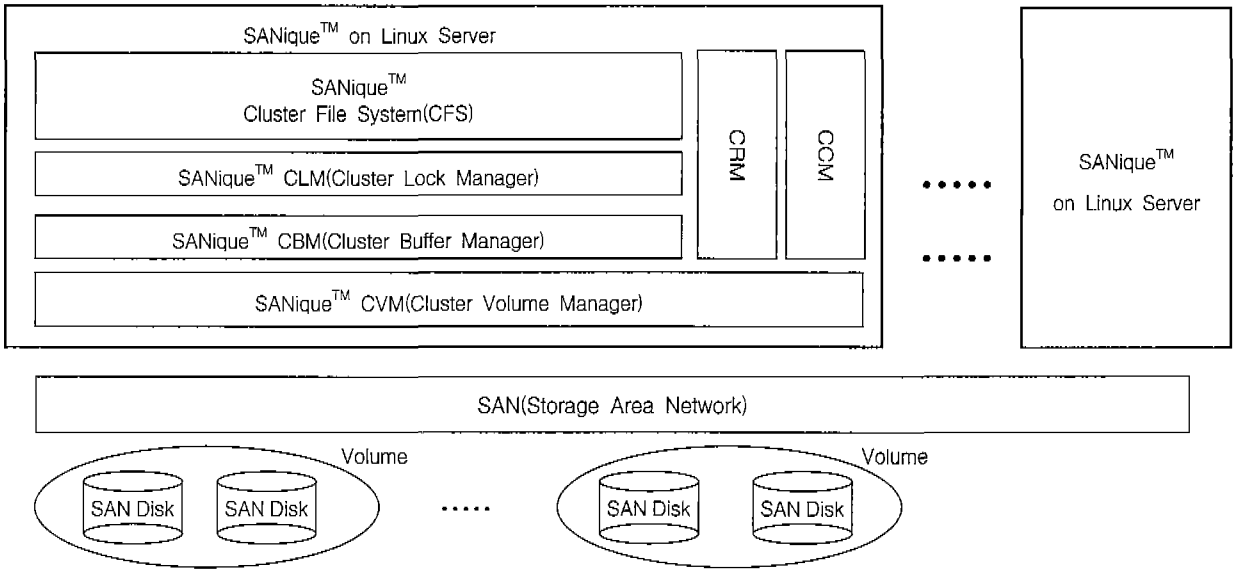
SAN 상에 연결된 리눅스 서버 클러스터들은 저장 장치 자원을 노드간에 공유한다. 이 공유는 다른 노드에 대한 데이타 서비스를 지원하지 않는 점에서 NFS의 공유와 구별될 수 있다. 즉, SANique™ CFS에 의해 클러스터의 모든 단일 노드들은 SAN에 직접 연결된 디스크 장치들에 대해 병행적 공유 접근이 가능하다. SANique™의 CFS는 64비트 주소 공간을 갖는 저널링(journaling) 화일 시스템이다. 단일 화일의 크기에 제한을 두지 않으므로, 최대 2^{64} 바이트 크기의 단일 화일을 생성할 수 있다. SAN에 직접 연결된 논리적 디스크들을 기반으로 형성되는 SANique™ CFS는 수퍼블록 영역, 비트맵 영역, 데이타 블록 영역으로 구성된다. 수퍼블록 영역은 화일 시스템의 데이타를 관리하기 위한 메타 데이타를 저장하고, 비트맵 영역은 디스크 공간의 할당 영역과 가용 영역을 관리하기 위한 영역이다. 전형적인 분산 화일 시스템에서는 화일 서버가 비트맵 영역을 관리하지만, 공유 화일 시스템 상에서는 여러 노드에 의해 관리 및 접근되므로 상호 배타적인 접근 방법이 필요하고, 비트맵 수정 연산에 대한 일관성 유지 방법이 필요하게 된다. SANique™ CFS에서는 비트맵 접근 단위를 세분화하여 다중 노드에 분산 시키므로 병렬성을 제공한 채 동시 접근 및 수정이 가능하다.

SANique™ CLM은 클러스터의 여러 노드에 의해 동시 접근되는 같은 화일에 대해 직렬성을 보장하기 위한 기능을 제공한다. 공유 볼륨 상의 화일 접근을 직렬화하고 일관된 데이타 접근을 보장하기 위하여 화일 수준의 병행수행 제어 방법을 이용한다. 이에 따라 같은 화일에 접근하는 모든 연산은 화일 단위의 데이타 일관성을 보장받는다.

SANique™ CBM 은 클러스터 노드들이 보유하고 있는 데이타 캐쉬 블록들이 다른 노드에서 필요할 때 전달될 수 있는 전역 버퍼 관리 기법(Global Buffer Management)을 이용한다. 기존 버퍼 관리 기법과는 달리, 데이타 블록을 디스크에서 읽게 되는 경우는 자기 자신의 버퍼에 해당 데이타 블록이 존재하지 않으며, 다른 노드의 버퍼에도 해당 데이타 블록이 없는 경우이므로 전역 버퍼 관리 기법인 상호협조 캐싱(cooperative caching) 기법을 활용한 전역 버퍼 관리 기법을 제공한다.

SANique™ CVM은 주어진 클러스터 내에 사용할 수 있는 모든 디스크 공간을 사용 가능한 가상 볼륨으로 구성하여 저장 풀(pool)을 형성하고 화일 시스템으로부터 사용될 수 있게 한다. 물리적 디스크 집합으로부터 논리적 디스크 영역을 구성하고 이를 화일 시스템의 주소 공간으로 전환할 수 있는 매핑 테이블을 유지한다. 또한 논리적 디스크는 물리적 디스크의 구성 방식에 따라, 스트라이핑, 미러링(RAID-1), 패러티 스트라이핑(RAID-5)등을 지원할 수 있다.

SANique™ CRM은 한 노드의 오류로 인하여 서비스 장애가 발생하는 경우 다른 노드에 영향을 주지 않도록 온라인 상에서 오류 노드를 복구하는 회복 관리기이다. 특히 범용 네트워크의 오류만 발생하고 SAN은 정상 작동하는 경우 다른



(그림 2) SANique™의 시스템 구성도

노드와의 제어 정보 교환 없이 디스크에 데이타를 기록하게 되어 메타 데이타 및 일반 사용자 데이타 블록에 불일치성을 발생할 수 있으므로, 회복 관리기에서는 이와 같은 오류 상황 또한 해결해야 한다. 또한 두 개 이상의 노드에서 서로 상대방 노드를 오류로 인식하여 발생하는 "split-brain"문제는 전형적인 분산 시스템의 문제이다. 공유 화일 시스템 상에서는 이러한 "split-brain" 문제가 화일의 일관성에 치명적인 문제를 발생시키며, 아직까지 명확한 해결방법이 제시되지 않고 있다. 본 논문의 다음 절에서는 이러한 오류 발생 상황과 문제점에 대해서 자세히 기술한다.

SANique™은 앞 장에서 기술한 바와 같이 공유 화일 시스템의 설계 방법인 대칭형과 비대칭형 시스템의 장점들을 활용하는 혼합 시스템이다. 즉, 서버 역할을 맡는 각 모듈들은 특정 한 노드에서 중앙집중적으로 수행되지 않게 하여 병목현상을 제거하였으며, 각 모듈에서 관리해야 하는 화일 시스템 정보를 적절히 분할하여 분산시키므로 중복 관리 정보가 발생하지 않게 하여 필요없는 네트워크 통신을 유발하지 않는다.

3. 공유 화일 시스템의 오류

본 절에서는 SANique™ 기반의 공유 화일 시스템상에서 발생하는 오류를 정의하고, 이들 오류가 발생시키는 문제점들을 온라인 상에서 해결하기 위한 회복 기법을 제공한다.

3.1 공유 화일 시스템의 오류 종류

공유 화일 시스템 상에서 발생하는 오류는 다음과 같이 분류될 수 있다.

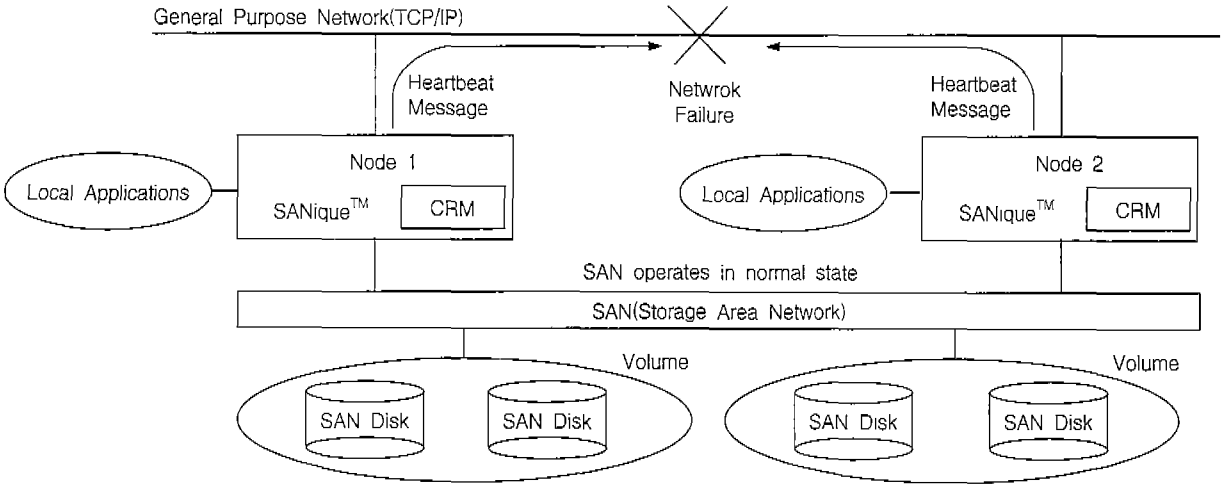
- 프로세스 오류
  공유 장치로 데이타 접근을 요청한 클라이언트 프로세스의 비정상적 종료에 의한 오류
- 시스템 오류
  정전 등에 따른 무전원 상태에 의한 오류, 시스템 하드웨어의 고장으로 인한 오류, 서버 소프트웨어의 비정상 종료에 의한 오류
- 장치 오류
  디스크 고장 또는 네트워크 장애로 인한 오류

위에 나열된 오류들은 일반적인 분산 시스템 또는 클러스터 시스템의 오류 탐지 방법 및 오류 회복 방법에 의해서 해결 가능하다. 특히 프로세스 오류와 디스크 오류는 독립형 시스템(stand-alone)의 오류 회복 방법의 확장형태로 오류를 회복할 수 있다. 그러나 클러스터 시스템의 네트워크 오류와 시스템 오류는 정확히 어떠한 오류 형태인지 탐지하기 힘들며 특히 클러스터 노드들이 오류로 인하여 그룹화 현상을 발생할 때 오류 노드 탐지 방법 및 회복 알고리즘의 구현은 더욱 어려워진다.

3.2 공유 화일 시스템의 "split-brain" 문제

대부분의 오류는 토큰(token) 기반의 주기적 메시지 전달 방법인 "heartbeat" 오류 탐지 방법에 의해 탐지된 후, 전형적인 오류 복구 방법에 의해 해결 가능하다. 그러나 네트워크 오류시, 서로 다른 노드에서 상대방 노드를 시스템 오류 발생으로 인지하는 "split-brain"문제가 발생한다. 상호 협조 체제(cooperative operation)로 수행되는 환경에서, 각 그룹이 기능적으로 완벽하지만, 서로 다른 그룹끼리 통신이 안되는 상황을 일반적인 "split-brain"문제라 정의한다[2]. (그림 3)은 SAN 기반의 공유 화일 시스템에서 발생하는 가장 일반적인 형태의 "split-brain"문제를 묘사하고 있다.

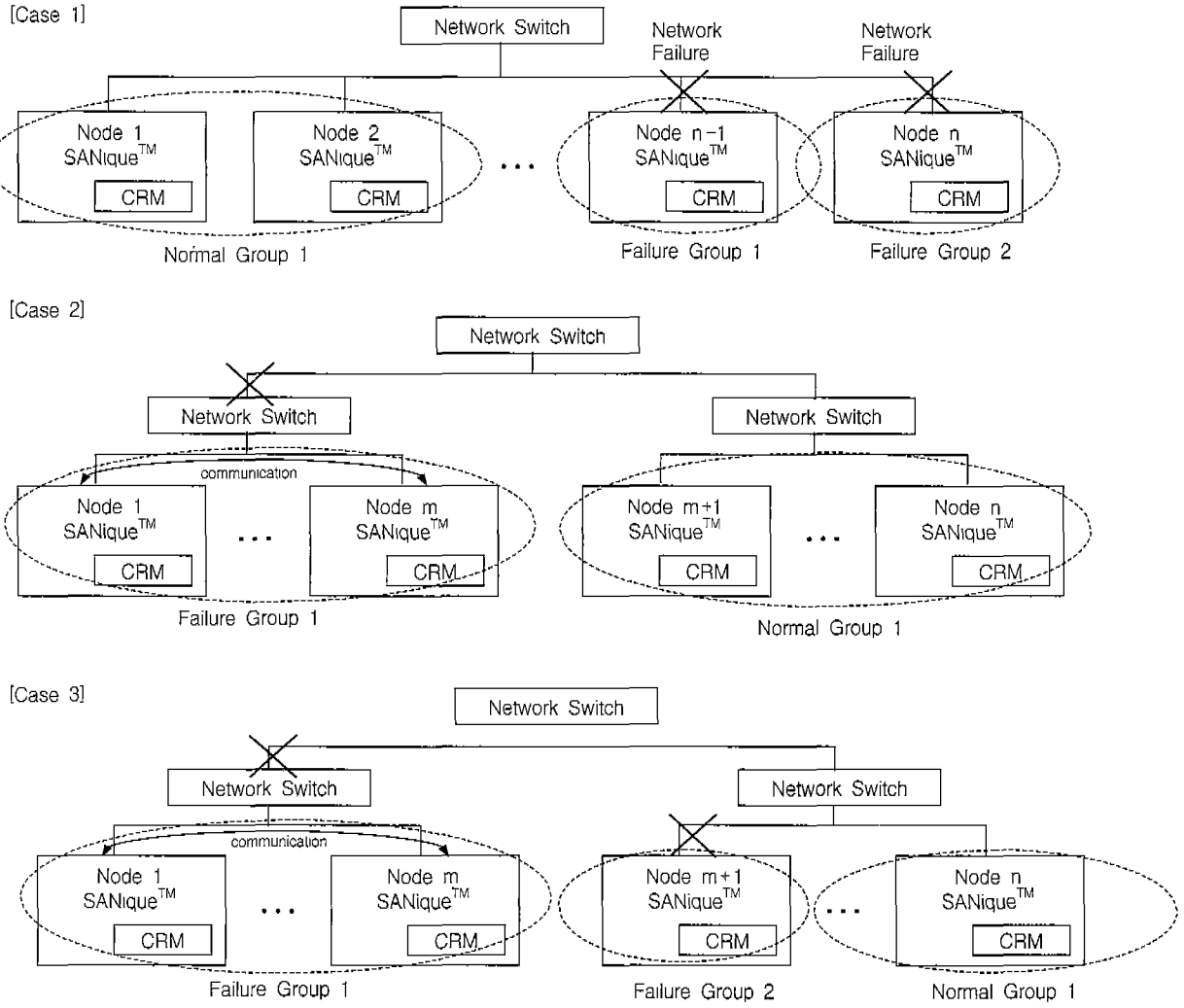
(그림 3)의 노드 1과 노드 2는 각각 주기적으로 상대 노드에 Heartbeat 메시지를 전송하여 상대노드의 상태를 점검한다. 주기적 점검 메시지는 서로간의 통신 수단인 범용 네트워



(그림 3) SAN 기반 공유 화일 시스템의 "split-brain" 문제

크를 통해 전달된다. 이 때 범용 네트워크의 오류가 발생하면 주기적 점검 메시지가 서로 전달되지 않게 되고, 각 노드에서는 서로 상대방 노드가 오류 상태인 것으로 판단하게 된다. 만약 실제로 노드 2가 시스템 오류로 인해 노드 자체가 다운된 경우라면 노드 1은 오류 회복을 위해 노드 1의 CRM을 통하여 노드 2에 대한 오류 복구 작업을 수행하게 되고, 그 후 노드 1에 마운트되어 있는 모든 화일 시스템에 대해서 정상적인 서비스를 재개할 수 있게 된다. 그러나 노드 2가 시스템 오류 상태가 아니라 단순한 쌍방간의 네트워크 오류인 경우에는 노드 2 역시 노드 1이 오류 상태에 있는 것으로 판단하고 노드 1에 대해서 오류 회복 작업을 진행하게 된다. 이와 같은 경우, 각각의 노드에서는 상대방에 대한 오류 회복 작업을 마친 후, 각각 자신의 지역적 응용을 수행하게 허가한다. 두 노드에서 화일 로킹이나 제어 정보 교환 없이 같은 화일 또는 같은 볼륨을 접근하게 되므로, 화일 시스템의 사용자 데이타 일관성은 물론 전체 공유 화일 시스템에 대한 관리 데이타에 대한 일관성까지 위반하게 되는 상황을 유발할 수 있다. 이와 같은 "split-brain"문제는 공유 화일 시스템의 전형적인 유형이라 할 수 있다.

SAN 기반 공유 화일 시스템에서 "split-brain" 문제는 네트워크 연결 유형에 따라 발생하는 상황이 다양해진다. (그림 4)는 공유 화일 시스템상에서 발생할 수 있는 split-brain 문제 상황을 일반화하여 설명하고 있다. 모든 클러스터가 하나의 네트워크 스위치 그룹에 연결되어 있는 경우 오류 그룹내의 노드 개수는 정확히 하나이다((그림 4)의 case 1). 또한 네트워크 오류가 한 곳 이상에서 발생하는 경우 다수의 오류 그룹이 발생 가능하다. (그림 4)의 [case 1]에서 정상 그룹은 반드시 1개가 존재하고, 그 그룹에 해당하는 노드들은 모두 서로 통신이 가능한 상태이다. 즉, 한 네트워크 장비로부터 연결된 클러스터이므로 네트워크 오류가 없는 노드들은 모두 서로 통신이 가능하다. 만약 [case 1]에서 정상 그룹이 한 개 이상 존재하는 경우라면, 그 노드들은 이중 네트워크 장치를 갖춘 노드들로서 또 다른 네트워크 장비를 통해 통신이 되는 그룹인 경우밖에 없다. 본 논문에서는 모든 노드가 하나의 네트워크 장비를 통해 통신한다고 가정한다. 따라서 (그림 4)의 [case 1]은 정상 그룹은 1개가 존재하며 그 그룹내의 노드는



(그림 4) SAN 기반 공유 화일 시스템에서 split-brain 의 일반적 형태

서로 통신이 가능한 상태이다.

또한 여러 개의 네트워크 스위치를 이용하여 클러스터를 형성한 경우 오류 그룹 내부에서도 서로 통신이 가능한 노드가 존재 가능하게 된다((그림 4)의 case 2). 이 경우 오류 그룹내에서 노드 1부터 노드 m 사이에는 서로 통신이 가능한 상태이며, 정상 그룹내부에서도 역시 노드 m+1부터 노드 n 사이에는 통신이 가능한 상태이다. 그러나 서로 다른 그룹간에는 통신이 불가능한 상태이다. (그림 4)의 [case 2]는 두 개의 그룹 내부에서 서로 상대방 그룹에 포함된 노드들이 오류 상태에 있다는 결정을 하게 되므로, 어느 그룹이 오류 그룹인지 한 그룹 내의 정보만으로는 판단할 수 없는 상태이다. 이와 같은 경우의 그룹화는 네트워크 장비 개수와 네트워크 구성에 따라 다수의 오류 그룹과 다수의 정상 그룹이 형성될 수 있다.

한 노드의 네트워크 오류와 네트워크 스위치 오류로 인한 노드 그룹의 네트워크 오류가 혼합하여 발생하는 경우([case 1]과 [case 2]의 혼합형태)에는 다수의 오류 그룹과 정상 노드 그룹이 형성될 수 있다((그림 4)의 case 3). 즉, 한 노드를 갖는 오류 그룹, 여러 노드를 갖는 오류 그룹, 정상 그룹으로 구성되는 형태이다.

(그림 4)를 통하여 SAN기반의 공유 화일 시스템에서 발생하는 "split-brain" 상황은 다음과 같은 어려움을 갖고 있음을 알 수 있다.

- 오류 그룹 내부에서 자신의 그룹이 오류 그룹인지 정상 그룹인지 판단 할 수 없다.
- 정상 그룹과 오류 그룹 모두 현재 오류 상황이 몇 개의 그룹으로 형성되었는지 파악할 수 없다.
- 네트워크 통신이 되지 않는 타 그룹들이 현재 네트워크 오류 상태인지 시스템 오류 상태인지 구별할 수 없다.
- 정상 그룹인 경우, 현재 자신 이외의 정상 그룹이 또 존재하는 지 알 수 없으며, 정상 그룹들 중에서 어떤 그룹이 가장 많은 노드를 포함하고 있는지 판단 할 수 없으므로, 최적의 정상 그룹으로 공유 화일 시스템 서비스를 재개할 수 없다.

이러한 문제는 오류 탐지시 상대 노드가 네트워크 오류인지 시스템 오류인지 정확하게 판단되지 않고 또한 그룹간 통신 장애로 인해 현재 오류 상황을 각각의 그룹에서 정확히 인지할 수 없어 발생하는 문제이다.

4. 공유 디스크 보드를 활용한 회복 기법

전형적인 "split-brain"문제를 해결하기 위해 여러 방법이 제시되었으나, 아직 완벽한 해결 방법이 제시되지 않았으며, 더욱이 공유 화일 시스템상에서 이러한 문제를 해결하기 위한 구체적인 알고리즘에 제시되지 않고 있다[2, 4]. 또한 기존 분산 환경에서는 고가용성을 목적으로 데이타 미러링(data mirroring) 기법의 클러스터 시스템이 대부분이므로, 이러한 환경에서 "split-brain"문제는 공유 화일 시스템의 문제보다 단순하다[3]. 즉, 분할된 노드간의 데이타가 동일한 상황이므로, 분할된 노드 중에서 어떤 노드가 회복 대상이 되고, 어떤 노드가 복구 후 재개되더라도 시스템 서비스에 큰 변화가 없다. 그러나 공유 화일 시스템에서는 최악의 경우 네트워크이 중단된 가장 작은 수의 노드를 갖는 그룹(그림 4의 [case 3]에서 오류 그룹 2)이 다른 모든 그룹의 노드들을 오류 처리하여 회복되고 화일 시스템 서비스를 재개하게 될 수 있으므로, 극도의 제한적인 즉, 화일 시스템 전체가 오류인 상태와 유사한 정도의 화일 시스템 서비스만을 제공할 수 밖에 없게 된다.

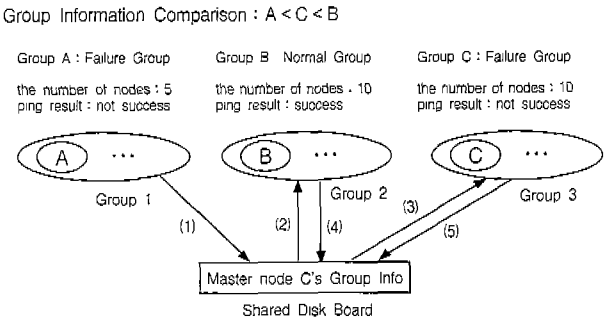
4.1 공유 디스크 보드와 그룹 정보 비교

SANique^{TM}의 CRM은 공유 화일 시스템에서 발생하는 "split-brain" 문제를 해결하기 위하여 클러스터내의 모든 노드가 네트워크 오류 시에도 SAN을 통하여 접근할 수 있는 SAN 디스크를 활용한다. 이 공간은 공유 디스크 보드(SDB : shared disk board)라는 이름으로 모든 클러스터 노드에서 이용된다.

SANique^{TM} CRM이 "split-brain"문제를 해결하기 위한 절차는 다음과 같다. 먼저, 토큰 기반의 "heartbeat"메시지 전송을 담당하는 탐지 데몬(daemon) 프로세스에 의해 초기의 노드-뷰(node-view)와 현재 노드-뷰가 달라지면 CRM에게 오류를 통지한다. 오류를 통지 받은 CRM은 현재 통신이 가능한 노드들을 그룹화하고, 이 그룹 중의 마스터 노드를 선정한다. 선정된 마스터 노드는 "split-brain"문제를 해결하기 위해 자신의 그룹에 대한 정보를 작성한다. 이 때 그룹 정보에는 두 가지 요소가 필수적으로 포함된다. 첫째, 현재 그룹의 통신 가능한 노드의 개수와 둘째, 오류 노드인지 정상 노드인지를 판별하기 위한 네트워크 상태 점검 결과가 포함된다. 네트워크 상태 점검은 외부로의 통신 상태를 점검하기 위한 다양한 유틸리티들이 이용될 수 있다. 이 결과를 앞서 설정한 공유 디스크 보드에 기록하고 이전에 설정된 다른 그룹의 그룹 정보가 있으면 읽어 온다. 읽어 온 정보가 자신의 그룹 정보보다 좋은 상황이면 전체 클러스터에서 제거(I/O fence out)된다. 그렇지 않으면 적정 시간 후, 공유 디스크 보드 기록 연산을 재수행한다. 재수행 후 자신의 그룹 정보보다 좋은 정보를 기록한 그룹이 없으면 "split-brain" 문제의 승자로 선정되고 자신의 그룹에 속한 노드를 제외한 모든 노드들에 대해 오류 회복 루틴을 수행한다. 이 때 그룹 정보간의 비교는 다음과 같이 수행된다. 먼저 두 그룹이 서로 다른 종류의 그룹인 경우, 즉 정상 그룹과 오류 그룹인 경우, 정상 그룹이 오류 그룹보다 비교우위에 있다. 그리고 같은 종류의 그룹인 경우에는 노드의 개수를 비교하여 노드 수가 많은 그룹이 비교우위에 있게 된다.

공유 디스크 보드에 기록하는 단계에서 또 다른 고려사항이 발생한다. 각 그룹의 마스터 노드들은 서로 네트워크 통신이 되지 않는 상태이므로 공통 접근 구역인 공유 디스크 보

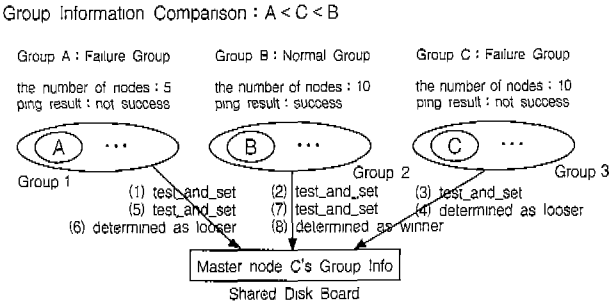
드에 기록, 판독 연산을 수행할 때 로킹 메커니즘이 제공될 수 없다. 예를 들어, 세 개의 마스터 노드 A, B, C가 있을 때, (그림 5)와 같이 각 그룹의 정보는 A < C < B의 비교우위가 있다고 가정하자. 이 때, 마스터 노드 A가 가장 먼저 공유 디스크 보드에서 정보를 읽고 아무도 정보를 기록하지 않았으므로 자신의 그룹 정보를 기록한다. 그 후, 마스터 노드 B가 A의 정보를 읽고 자신의 그룹 정보가 그룹 A의 그룹 정보보다 더 우수하므로 그룹 B의 그룹 정보를 공유 디스크 보드에 기록하게 된다. 그러나 그 사이에 즉 마스터 노드 B가 A의 그룹 정보를 읽어간 후 아직 기록하기 전에((그림 5)의 순서 (3)), 마스터 노드 C가 마스터 노드 A의 정보를 읽어가고 역시 자신의 그룹 정보보다 좋지 않는 상황이므로 기록을 결정한다. 이 때, B가 먼저 기록하고 C가 후에 기록하게 되면, 마스터 노드 B가 마스터 노드 C보다 더 좋은 그룹 정보를 가지고 있음에도 불구하고 B의 그룹 정보는 손실되게 된다. 결국 이 예에서 세 개의 그룹중에서 그룹 C의 오류 그룹이 승자가 되어 다른 노드들을 클러스터 내에서 제거하게 되고, 이 노드들에 대한 오류 회복 작업 후에 공유 화일 시스템 서비스를 재개한다. 그러나 그룹 C는 오류 그룹, 즉 외부와 네트워크 통신이 안되는 그룹이므로 공유 화일 시스템 서비스를 외부 클라이언트들에게 제공하지 못하고 내부 응용에만 제공하는 제한을 받게 된다. 이 경우 그룹 B가 "split-brain" 문제의 승자가 된다면 더 폭 넓은 공유 화일 시스템 서비스를 제공할 수 있다.



(그림 5) 공유 디스크 보드의 기록 수행의 문제

이것은 각 노드간의 통신이 되지 않으므로 공유 데이타 접근에 대해서 적절한 병행수행 제어를 제공하지 못하게 되어 발생하는 문제이다. 이를 해결하기 위해서는 저장 장치 자체에서 제공되는 로킹인 디바이스 로크(device lock)[9, 10]를 사용해야 한다. 그러나 디바이스 로크는 디스크 자체에 부가되는 로크이므로 일반 SAN 디스크 상에서 활용하기에 어려움이 있다. 따라서 제안하는 방법에서는 test-and-set 루틴을 원자적(atomic)으로 구현하여 판독과 기록을 동시에 수행한다. 그 후에 읽어간 판독한 정보를 비교하여 우위인 경우에만 공유 디스크 보드에 test-and-set 루틴을 재수행하게 된다. 즉, (그림 5)의 예에서 보인 순서와 동일하게 test-and- set 루틴을 적용하면 다음 (그림 6)과 같다.

마스터 노드 A가 먼저 test-and-set 루틴을 수행하고 자신의 그룹 정보를 기록한다. 마스터 노드 A가 판독한 정보는 공유 디스크 보드의 초기 값이므로 적정 시간 후 test-and-set를 재수행하게 된다((그림 6)의 (5)). 마스터 노드 B는 자신의 그룹 정보를 기록하고 A의 그룹 정보를 읽어 간다. 마스터 노드 C 또한 B의 그룹 정보를 읽어 가고 자신의 그룹 정보를 기록한다. 이 상황에서 test-and-set를 재수행할 수 있는 마스터 노드는 A와 B가 된다. 즉, 노드 C는 B가 기록한 정보를 읽은 후, 비교우위에 있지 않으므로 test_and_set 수행을 포기하게 된다. 따라서 계속하여 test_and_set을 재수행 후 마스터 노드 B 만이 자신의 정보를 기록하게 되어 마지막 승자가 된다.



(그림 6) test-and-set을 활용한 공유 디스크 보드 기록 방법

4.2 SANique™ CRM의 회복 알고리즘

앞 절에서 살펴본 바와 같이, 본 논문에서 제시하는 회복 알고리즘은 각 그룹간의 그룹 정보를 공유 디스크 보드를 활용하여 교환하고 승자가 되는 그룹이 모든 오류 그룹의 노드들에 대해 회복절차를 수행하고 패자가 되는 그룹은 리눅스 클러스터 화일 시스템으로부터 제거되는 것이다. 본 회복 방법에 대한 의사 코드는 다음과 같다.

Phase 1 : Detection

1. 주기적인 "heartbeat" 메시지를 브로드캐스트 방식으로 전송한다.
2. if (current Node-View != Previous Node-View) SANique™ CRM에 오류 노드 리스트를 전달하고 오류 상황임을 알린다.

Phase 2 : Solve the Split-Brain Problem

1. 자신의 그룹에서 네트워크 통신이 가능한 노드 리스트를 그룹화하고, 마스터 노드를 선정한다.
2. 자신의 그룹에 대한 그룹 정보를 작성한다.
   2-1 네트워크 상태 점검 결과 작성
   2-2 자신의 그룹의 노드 개수 작성
3. 공유 디스크 보드를 open한다.
4. 공유 디스크 보드에 그룹 정보를 test-and-set 루틴을 이용하여 기록한다.
5. for (i = 0 ; i < Given Trial No Parameter ; i ++) {
   5-1 if (my group information > fetched group information)
   5-1.1 sleep(Given Time) ;
   5-2 else /* Looser Group */
   5-2.1 I/O Fence out 루틴을 수행하고 자신의 그

룹은 공유 화일 시스템 그룹에서 제거된다 (Looser Group).

5-2.2 회복 알고리즘을 종료한다.

6. If (fetched group information == my group information)

   6-1 Winner = my group

Phase 3 : Recovery(Only Winner Group)

7. 자신의 그룹 노드를 제외한 모든 노드들에 대해 회복 루틴을 시작한다.
8. SANique™ CRM이 모든 서브 시스템 모듈의 회복 작업에 대한 coordination을 제어한다.
9. 오류에 대해 회복이 끝나고 현 그룹의 노드들로 공유 화일 시스템 서비스를 재개한다.

제시된 회복 알고리즘에 의해 다중 노드의 오류 발생시에도 오류 그룹들의 회복 작업 혼란없이 정상 그룹 중에서 가장 우수한 그룹을 선정하여 온라인 상에서 회복할 수 있게 되며, 오류 상태에서 가장 적절한 화일 시스템 서비스를 재개할 수 있게 된다.

5. 결론 및 향후 연구

본 논문에서는 리눅스 클러스터 기반의 SAN 기반 공유 화일 시스템인 SANique™에 대한 시스템 구성도 및 설계 방법을 기술하였으며, 구현된 공유 화일 시스템 상에서 발생 가능한 노드 오류 상황을 조사하고 이의 해결방안을 제시하였다. 특히 오류 상황 중에서 "split-brain"문제를 해결하기 위하여 공유 디스크 보드라는 SAN 디스크의 공통적인 논리 볼륨을 활용하여 가장 우수한 노드들로 구성된 정상 상태의 노드들을 그룹화하여 온라인 상태에서 노드들의 오류를 회복하고 서비스를 재개할 수 있는 회복 알고리즘을 제시하였다. 본 방법을 통하여 공유 화일 시스템상의 오류를 쉽게 탐지할 수 있으며, 오류 노드 제거에 대한 혼란 없이 최적의 그룹으로 화일 시스템 서비스를 제공할 수 있게 된다.

현재 제안된 방법은 리눅스 클러스터의 노드 개수가 많아지는 경우, "split-brain"문제를 해결하는 시간이 증가하게 된다. 특히, 클러스터의 노드 개수가 많아짐에 따라 오류 그룹과 정상 그룹으로 분할되는 그룹수가 많아지게 되면 분할 된 그룹 중에서 최적의 그룹을 판별하기 어려워진다. 따라서 현재 노드의 수가 100개 이상인 클러스터 상에서도 "split-brain" 문제를 현실적인 시간 안에 해결하기 위한 우선순위 기반의 오류 탐지 및 회복 방법을 계속 진행중이다.

참 고 문 헌

[1] Sang G. Oh, and Jang S. Lee, "SANique™ : A SAN File system for Linux Cluster," Technical White Paper - Draft, MacroImpact. Co. Ltd., 2001.

[2] C. C. Fan and J. Bruck, "The Raincore Distributed Session Service for Networking Elements," Proc. Of the International Parallel and Distributed Processing Symposium, 2001.

[3] P. T. murray, R. A. Fleming, P. D. Harry, P. A. Vickers, "Somersault : Enabling Fault-Tolerant Distributed Software Systems," Technical Paper HPL-98-81, Internet Comm. Systems Dept, Hewlett-Packard Labs. Bristol, 1998.

[4] P. S Weygant, "Primer on Clusters for High Availability," Technical Paper at Hewlett-Packard Labs, CA, 2000.

[5] R. Sandberg, D. Goldberg, S. Kleiman, D. Walsh, and B. Lyon, "Design and Implementation of the Sun Network File Systems," Proc. Of the Summer USENIX Conf. 1985.

[6] U. Vahalia, Unix Internals : The New Frontiers, Prentice-Hall, NJ, 1996.

[7] M. D. Dahlin, "Severless Network File Systems," Ph. D. Thesis at Computer Science Graduate Division of University of California at Berkely, 1995.

[8] Oracle Parallel Server, An Oracle Technical White Paper, November, 1998.

[9] S. R. Soltis, T. M. Ruwart, and M. T. O'keefe, "The Global File Systems," Proc. Of the 5th NASA Goddard Conference on Mass Storage Systems and Technologies, 1996.

[10] K. W. Preslan, A. Barry, J. Brassow, R. Cattelan, A. Manthei, E. Nygaard, S. Oort, D. Teigland, M. Tilstra, and M. O'keefe, "Implementing Journaling in a Linux Shared Disk File System," Proc. Of the 8th NASA Goddard Conference on Mass Storage Systems and Technologies, 1999.

[11] K. W. Preslan, A. Barry, J. Brassow, R. Cattelan, A. Manthei, B. Marzinski, E. Nygaard, S. Oort, D. Teigland, M. Tilstra, S. Whitehouse and M. O'keefe, "Scalability and Failure Recovery in a Linux Cluster File System," Proc of the 4th Linux Showcase and Conference, 2000.

[12] M. Satyanarayanan, "Scalable, Secure, and Highly Available Distributed File Access," IEEE Computer, 1990.

이 규 웅

email : lockw@mail.sangji.ac.kr

1990년 한국외국어대학교 전자계산학과 (이학사)

1992년 서강대학교 대학원 전자계산학과 (공학석사)

1998년 서강대학교 대학원 전자계산학과 (공학박사)

1998년~2000년 한국전자통신연구원 인터넷서비스 연구부 선임연구원

2000년~현재 상지대학교 컴퓨터·정보공학부 전임강사

관심분야 : 트랜잭션 처리, SAN 기반 자료저장 시스템, 분산 및 실시간 DB