SAN 기반 클러스터 공유 화일 시스템 SANique™의 오류 노드 탐지 및 회복 기법

이규웅*

Detection and Recovery of Failure Node in SAN-based Cluster Shared File System SANique™

Kyu-Woong Lee*

이 논문은 2008년도 상지대학교 교내연구비 지원에 의한 결과임

요 약

본 논문은 저장장치 전용 네트워크인 SAN 상에서 운영되는 공유 파일 시스템 SANique™의 개괄적인 설계 방법과 공유 파일 시스템 내의 오류 노드 탐지 및 회복 기법에 대한 방법을 설명한다. SAN 기반 공유 파일 시스템의 특징 및 구조를 설명하고 SANique™의 구성 요소와 개괄적 설계방법을 기술한다. 또한, 공유 파일 시스템에 참여하고 있는 컴퓨팅 노드의 오류로 인한 서비스 지연 또는 중지를 방지하기 위하여 오류 노드 탐지 및 회복 기법을 설명한다. 대규모 컴퓨팅 노드로 구성된 공유 파일 시스템 상에서 발생할 수 있는 오류의 종류를 나열하고, 오류로 인한 분할된 서브 그룹들 간의 오류 상황을 상호 탐지 할 수 있는 방법을 설명하고 이를 해결하기 위한 기법을 제안한다.

ABSTRACT

This paper describes the design overview of shared file system SANique™ and proposes the method for detection of failure node and recovery management algorithm. We also illustrate the characteristics and system architecture of shared file system based on SAN. In order to provide uninterrupted service, the detection and recovery methods are proposed under the all possible system failures and natural disasters. The various kinds of system failures and disasters are characterized and then the detection and recovery method are proposed in each disconnected computing node group.

키워드

SAN, 공유 파일 시스템, 시스템 오류, 탐지 및 회복

Key word

SAN, Shared File System, System Failure, Failure Detection and Recovery

* 상지대학교 컴퓨터정보공학부

접수일자 : 2009. 08. 20
심사완료일자 : 2009. 10. 18

Ⅰ. 서 론

사용자 참여 멀티미디어 콘텐츠 제작 및 제공이 급속도로 증가함에 따라 다양한 유형의 대규모 데이터를 원활하게 서비스하기 위한 컴퓨팅 플랫폼의 변화를 요구하고 있다. 데이터 중심의 대규모 웹 서비스를 제공하기 위해 클러스터 기반의 파일 시스템이 필요하게 되었으며, 대규모의 데이터들을 효율적으로 처리하기 위한 분산 데이터 저장 및 검색 방법이 필요하게 되었다. 다량의 데이터를 신속하게 처리할 필요성이 높아짐에 따라 대용량 데이터를 저장하고 처리하기 위한 플랫폼에 대한 요구가 높아지고 있다. 대표적인 클러스터 파일 시스템의 구조는 상호 연결된 컴퓨팅 노드 그룹이 화이버 채널과 같은 저장장치 전용 네트워크인 SAN상에 연결된 저장공간을 공유 할 수 있도록 하는 공유 파일 시스템을 제공하는 구조이다. 본 논문은 위와 같은 클러스터 공유 파일 시스템 상에서 발생하는 시스템 오류 및 자연적 재해 상황을 탐지하는 기법을 설명하고 오류를 처리하여 서비스를 즉각 재개 할 수 있는 오류 회복 기법에 대해 기술한다.

본 논문의 구성은 다음과 같다. 제2장에서 클러스터 공유 파일 시스템의 특징 및 기존 관련 연구를 기술하고 제3장에서 공유 파일 시스템 상에서 발생하는 오류의 유형을 정의하고 이를 탐지하기 위한 방법을 설명한다. 탐지된 오류 유형들을 분석하여 이를 해결하고 서비스를 재개할 수 있는 방법을 제4장에서 기술하고 마지막으로 제5장에서 결론을 맺는다.

Ⅱ. 연구 배경 및 관련 연구

가. 클러스터 공유 파일 시스템의 관련 연구

최근 저장 장치 구조에 대한 클러스터링에 대한 연구가 급증하고 있으며 이러한 환경의 클러스터 파일 시스템 및 시스템 소프트웨어의 상용화가 증가하고 있다[1,2,3]. 클러스터 파일 시스템은 SAN 기술의 발전으로 최근 많은 연구가 진행되고 있으며, 고전적인 SAN 기반 클러스터 파일 시스템으로는 미네소타 대학에서 초기 개발을 시작한 GFS(Global File System), 버클리 대학의 xFS, 카네기 멜론 대학의 NASD(Network Attached Secure Disk) 기반 분산 파일 시스템 등이 초기 클러스터 파일 시스템의 원형으로 다양한 상용제품의 기반 기술로 활용되고 있다[1,2,4]. 최근 분산 파일 시스템은 구글 파일 시스템 GFS(Google File System) [5,6,7], 한국전자통신연구원의 OASIS[8]와 같이, 다양한 사용자의 대규모 데이터 웹 서비스 요구에 맞추기 위해 각 서버들을 클러스터링 하여 하나의 대용량 파일 시스템으로 형성하는 객체 기반 파일 시스템 및 지능형 클러스터 파일 시스템으로 확장 개발되고 있다. 구글 클러스터 파일 시스템은 구글의 동영상 검색 및 저장 서비스, 이미지 서비스, 구글 어스 등과 같은 구글의 모든 데이터 집중적인 웹 서비스를 제공하기 위한 구글 플랫폼으로 사용되는 대표적인 분산 클러스터 파일 시스템이다[5,6,7].

나. SANique™ 클러스터 파일 시스템의 구조

최근의 구글 클러스터 파일 시스템 등은 구글 웹 서비스 또는 구글의 목적화된 응용 등과 같이 특정 목적을 전제로 서비스되는 반면 SANiqueTM은 기존의 서버들과 마찬가지로 범용적 목적의 완벽한 응용 서비스를 제공하기 위한 클러스터 공유 파일 시스템이다. SANiqueTM은 클러스터 파일 시스템의 각 컴퓨팅 서버간 진정한 공유를 제공하기 위하여 그림 1과 같은 시스템 구조를 갖는다. 각 서버들은 기존 파일 시스템위에 SANiqueTM의 주요 구성 모듈인 CFS와 CVM을 탑재하여, 공유 디스크 풀에 존재하는 모든 파일들에 대해 읽기/쓰기가 가능하게 되며, 그 연산의 결과는 즉각적으로 클러스터 내 모든 노드들에 의해 공유되어 질 수 있다. 그림 1의 첫 번째, 두 번째 서버에 의한 쓰기 연산은 클러스터 내의 공유 디스크 공간으로 반영되어, 세 번째 및 네 번째 서버에 의해 다른 서버의 간섭없이 각각 읽기가 가능하다.

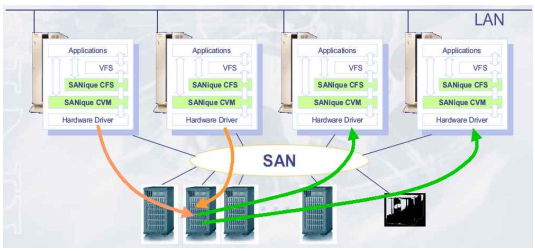

그림 1. SANique™의 시스템 구조도
Fig. 1. System Architecture of SANique™

CFS의 주요 파일 시스템 정보는 특정 노드에 할당되지 않고 클러스터를 구성하는 모든 서버들에 분할 저장되므로 공유 디스크를 접근할 때 마다 각 서버들의 정보 공유를 통해 디스크 접근을 수행하게 된다. 이러한 구조에 의해 특정 서버 오류로 인한 전체적 서비스 중단 현상을 방지 할 수 있으며, 빈번하게 접근되는 특정 데이터에 대해서도 디스크 접근에 대한 병목현상을 유발하지 않게 된다. CVM은 클러스터 볼륨 관리기로서 SAN에 부착된 모든 디스크들을 CFS에 의해 관리되게 하기 위하여 공유 볼륨으로 구성하는 볼륨 관리기이다. 논리적 볼륨은 물리적 디스크의 구성 방식에 따라 스트라이핑, 미러링, 패러티 스트라이핑 등 다양한 형식으로 지원될 수 있다.

Ⅲ. 클러스터 파일 시스템의 오류 탐지 기법

가. 오류 탐지 프로세스 구조도

SANique™ 시스템은 회복 탐지 및 복구를 위해서 서버 프로세스들을 감시하고, 회복 작업을 지시하는 고가용성 서비스 계층을 그림 2와 같이 서버 프로세스의 상위 계층에서 운영한다.

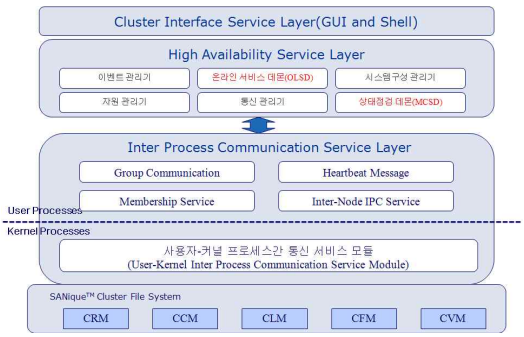

그림 2. 오류 탐지 프로세스 구조도
Fig. 2. Process Architecture of Failure Detection

고가용성 서비스 계층의 온라인 서비스 데몬(OLSD)은 주변의 노드에 대한 상태를 점검하여 특정 노드가 온라인 서비스를 제공하고 있는지에 대한 상태 점검과 자신 노드의 서비스 프로세스들이 정상적인 수행을 하고 있는지 자체 검사 작업을 수행한다. 특히, 토큰 기반의 주기적 메시지 전달 방법에 의해 주변의 노드 구성이 이전상태와 달라지는 경우 온라인 서비스 데몬에 이를 통지하고, 온라인 서비스 데몬은 해당 노드의 상태관리 데몬(MCSD)에게 상태관리를 위한 통신을 요청한다. 해당 노드의 상태관리 데몬이 응답하지 않는 경우나 또는 그 노드의 일부 자원 즉, 그림 2에 나타난 커널 프로세스 및 시스템 프로세스들에 대한 상태가 올바르지 않은 경우 전체 클러스터 구성에서 그 노드를 제거하고, 그 노드의 기능을 다른 노드에게 전이하는 회복작업을 수행하게 된다.

나. 오류 탐지 프로세스 수행 절차

오류 탐지 절차는 그림 3과 같이 오류 탐지 노드 Ni가 오류 예상 노드 Nj에 대해 상태점검 데몬과의 통신을 통해 이루어지게 된다.

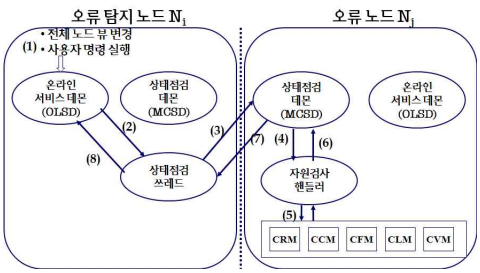

그림 3. 오류 탐지 데몬의 수행 절차
Fig. 3. Flow of Daemon for Failure Detection

오류 탐지 첫 단계에서는 주기적 메시지에 의한 노드 뷰 변경을 감지하여 온라인 서비스 데몬에 통지하는 단계이며, 둘째 단계는 그림 3의 (2),(3)에 나타난 것처럼 온라인 서비스 데몬에 의해 상태점검 쓰레드를 생성하고, 오류가 예상되는 노드 Nj의 상태점검 데몬과의 통신을 요청하는 단계이다. 세 번째 단계인 그림 3의 (4)~(7)은 노드 Nj의 상태점검 데몬이 자신의 노드의 자원, 즉 커널 및 시스템 프로세스들을 점검하고 이를 통지하는 단계이고, 마지막으로 그림 3의 (8)에서와 같이 노드 Ni의 상태점검 쓰레드는 노드 Nj의 오류 여부를 판단하여 해당 노드 Nj를 클러스터에서 강제 제거하고 그 기능을 전이받는 회복 작업을 클러스터 회복관리기인 CRM에 지시하는 단계이다. 오류 판단이 완료되면 모든 오류 회복절차는 노드 Ni의 CRM을 통하여 수행하게 된다.

오류 노드 Nj가 전원오류나 완전한 시스템 정지상태인 경우, Nj의 상태점검 데몬은 물론 어떠한 프로세스도

존재하지 않으므로 노드 Ni의 상태점검 쓰레드는 통신 불가로 인해 Nj의 오류를 결정하게 된다. 그러나 노드 Nj의 다른 종류의 오류나 프로세스 오류인 경우 Nj의 상태점검 데몬은 자신의 노드의 자원들을 상태 점검하여 이를 통보한다.

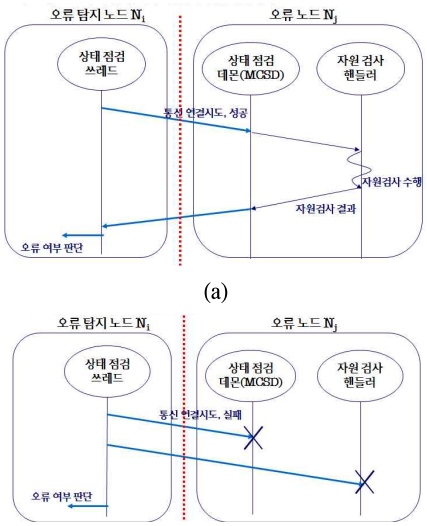

그림 4. 상태 점검 데몬의 통신 흐름도
(a) 오류노드가 자원검사 수행결과 전송하는 경우
(b) 오류노드가 시스템 오류인 경우
Fig. 4. Flow of MCSD
(a) In case of MCSD sends the check results
(b) In case of both MCSD and handler are failure

또한 노드 Nj의 상태점검 데몬의 단순 종료로 인해 노드 전체가 오류로 처리되는 것을 방지하기 위해 노드 Ni는 노드 Nj의 상태점검 데몬과 통신에 실패하게 되는 경우 노드 Nj의 온라인 서비스 데몬과 통신을 재시도하게 되어 이중통신을 통한 오류 탐지를 수행한다. 오류 탐지 노드 Ni의 상태점검 쓰레드와 오류 노드 Nj의 상태점검 데몬 및 온라인 서비스 데몬과의 통신 흐름을 도식화하면 그림 4와 같다. 그림 4의 (a)에 해당하는 통신 흐름도는 노드 Ni의 상태점검 쓰레드의 요청에 따라 오류 노드 Nj의 상태점검 데몬이 해당 노드의 모든 자원들, 즉 CFM, CLM 등과 같은 시스템 모듈 및 서버 프로세스들을 점검하여 그 결과를 전송하고 노드 Ni는 노드 Nj의 오류 여부를 판단하게 된다.

그림 4의 통신 흐름도 (b)는 노드 Nj가 완전한 시스템 오류를 발생하여 어떠한 네트워크 연결도 불가능한 상태를 보이고 있다. 그림 5의 통신 흐름도 (a)은 단순한 상태점검 데몬의 오류로 인해 전체 시스템이 오류로 처리되는 것을 방지하기 위하여 노드 Nj의 온라인 서비스 데몬과 중복점검을 시도한 후, 상태점검 데몬을 재수행하게 하여 그림 4의 (a)와 같은 통신 흐름을 재개하여 노드 Nj의 오류 점검 결과를 전송하는 통신과정을 보이고 있다.

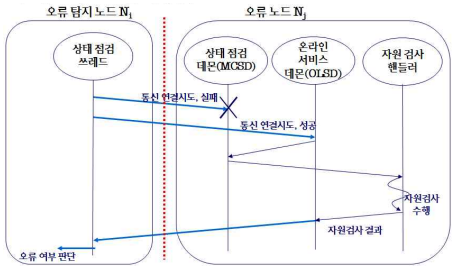

그림 5. 상태점검 데몬 오류 통신 흐름도
(a) 오류노드의 MCSD만 오류인 경우
Fig. 5 Flow in case of MCSD Failure
(a) In case of MCSD is failure

다. 분할된 오류 노드 그룹 유형 분석

클러스터 공유 파일 시스템에 네트워크에 의한 오류가 발생한 경우에는 서로 다른 노드에서 상대방 노드를 시스템 오류 노드로 인지하게 되는 분할 오류 노드 그룹 문제(split-brain problem)가 발생하게 된다. 클러스터 공유 파일 시스템에서 네트워크 단절로 인해 여러 그룹으로 분할되고 각 분할 그룹이 기능적으로 완벽하지만, 서로 통신이 단절된 상황을 클러스터 환경의 분할 오류 노드 그룹이라 정의한다. 그림 6은 두 개의 노드로 구성되는 클러스터 파일 시스템에서 발생하는 분할 오류 노드 그룹 문제를 보이고 있다.

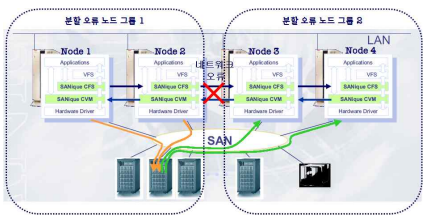

그림 6. 분할 오류 노드 그룹 문제
Fig. 6. Split-Brain Failure Node Group

그림 6의 분할 오류 노드 그룹 1에 속해 있는 노드 1과 노드 2는 상호 통신이 가능하므로 온라인 서비스 데몬 및 상태 점검 데몬이 상호 완벽하게 수행된다. 그러나 분할 오류 노드 그룹 2의 노드들과는 통신이 단절된 상태이므로 노드 3과 노드 4의 자원들에 대해 그림 4의 (2)의 통신 흐름도에 따라 오류 보고를 받게 된다. 마찬가지로 분할 오류 노드 그룹 2에서도 노드 1과 노드 2에 대해 오류 보고를 받게되므로 그룹 1과 그룹 2는 각각 상대 그룹에 대해 오류가 있음을 알게 된다. 이 때, 오류 처리의 전형적인 방법에 의해서 각각의 그룹은 상대 오류 노드의 모든 업무를 물려받고(failover) 파일 시스템 서비스를 재개하려고 시도하게 된다. 그러나 하나의 클러스터 파일 시스템내에 두 개의 서비스가 존재하면 파일 시스템의 논리적 오류가 발생하게 되므로 하나의 그룹은 서비스를 포기하고 패닉상태로 접어들어야 한다. 분할 오류 노드 그룹 문제는 그림 6과 같이 통신이 단절된 상태에서 각 그룹이 오류 노드 그룹인지 서비스를 재개해야 할 그룹인지 판단하는 문제이다. 분할 오류 노드 그룹은 네트워크의 연결 유형에 따라 다양한 형태로 분류될 수 있다. 연결된 네트워크의 계층적 위치에 따라서 오류 그룹이 분할 될 수 있으며, 네트워크 카드의 오류 등으로 인하여 단일 노드의 고립상태를 만드는 오류가 발생 될 수 있다. 그림 7은 전형적인 네트워크 오류로 인한 분할 오류 노드 그룹의 예이다. 각 노드의 네트워크 장비 고장으로 인한 오류는 단일 오류 노드로 구성되는 오류그룹을 형성한다.

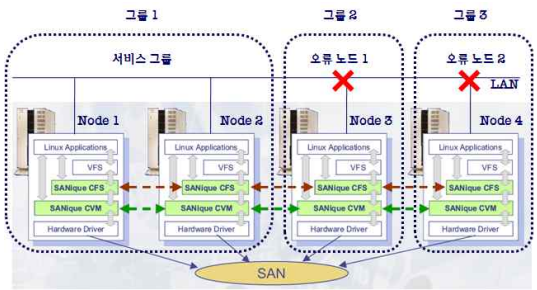

그림 7. 전형적 분할 오류 노드 그룹의 예
Fig. 7 Example of Split-Brain Failure Groups

네트워크의 장비가 계층적으로 구성된 클러스터에서는 좀 더 다양한 분할 그룹이 형성될 수 있다. 그림 8에서 노드 3과 노드 4는 인접한 상단의 네트워크 스위치 까지는 통신이 가능하므로 서로 통신이 가능하다. 그러나 네트워크 스위치 윗단에서 통신 오류가 발생하여 노드 1과 노드 2의 통신이 불가능하여 자신의 그룹(그룹 2)이 클러스터내에서 유일한 그룹인 것으로 판단가능한 상황이다. 유사하게 노드1과 노드 2는 하나의 그룹(그룹 1)으로 형성되어 그룹 2에 포함된 모든 노드를 오류 노드로 인식하게 된다. 각 그룹은 상대 노드를 클러스터내에서 제거하고자 하는 불일치한 오류 노드 인식 상황이 발생한다.

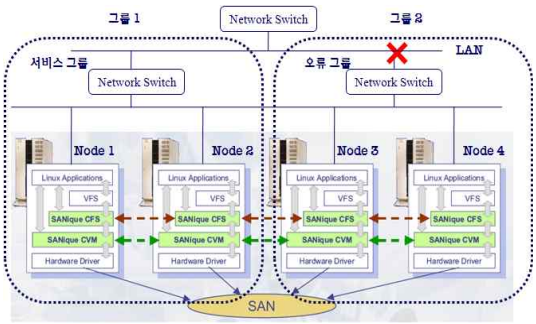

그림 8. 계층적 구조의 분할 오류 노드 그룹
Fig. 8. Split-Brain Groups in Hierarchical Structure

다양한 오류 노드 그룹의 예에서 다음과 같은 문제점을 갖고 있다. 첫째, 오류 그룹 자신이 오류 그룹인지 정상 그룹인지를 판별하기 어려우며, 둘째, 클러스터 내의 노드들이 몇 개의 오류 그룹으로 분할되었는지 판단할 수 없다. 셋째, 통신이 되지 않는 노드가 시스템 오류인지, 아니면 단순한 네트워크 오류인지 판단할 수 없다. 마지막으로, 분할그룹 중에서 가장 많은 노드로 구성된 그룹을 선별할 수 없어, 최적의 그룹으로 서비스를 재개하기 어렵다. 위와 같은 문제들은 시스템 오류와 네트워크 오류가 정확히 탐지되지 않아 발생하며, 또한 분할그룹간의 네트워크 통신이 단절되어 각 그룹의 상황을 서로 정확히 인지할 수 없어서 발생하는 문제이다.

Ⅳ. 분할 오류 노드 그룹의 회복기법

전형적인 분할그룹 문제를 해결하기 위해 여러 기법들이 제시되었으나, 최적의 해결방법이 아직 제시되지 않았으며, 더구나 클러스터 파일 시스템 상에서의 분할

그룹 문제에 대한 구체적인 방법은 아직 연구되지 않고 있다[9]. 기존의 분산 환경에서 이중화를 위한 미러링(mirroring) 기법은 다수의 노드가 주 서버와 같은 기능, 같은 데이터를 보관하고 있으므로, 대체 서버를 선택하는 데 있어서 큰 어려움이 없어, 본 연구의 기반 환경이 되는 클러스터 파일 시스템에서 보다 단순하게 해결할 수 있다[10]. 그러나 클러스터 파일 시스템 환경에서는 분할된 오류 노드 그룹 중에서 최적의 그룹을 판별하여 다른 모든 그룹의 서비스를 대체하고 오류 처리 할 수 있는 기법이 필요하지만, 통신 단절로 인해 분할그룹의 개수와 그룹 내의 노드 정보를 판단할 수 없는 근본적인 문제를 갖고 있다.

가. 공유 디스크를 활용한 분할 오류그룹 비교 기법

회복 관리기 CRM에서는 분할그룹 문제를 해결하기 위하여 공유 디스크를 활용한다. 저장장치 전용 네트워크인 SAN에 직접 연결된 공유 디스크를 활용하는 기법을 본 논문에서 제안한다. 이 공유 디스크 공간은 SDB라는 이름으로 모든 노드들이 마운트하여 사용하게 된다. 회복 관리기 CRM은 다음과 같은 절차로 수행된다.

(1) 오류 탐지 단계 : 먼저 토큰 기반의 주기적 점검 메시지를 담당하는 오류 탐지 데몬 프로세스에 의해 초기에 구성된 클러스터 뷰와 현재 노드 뷰가 달라지면 회복 관리기 CRM에게 오류 탐지를 통보한다.

(2) 그룹 형성 및 마스터 노드 선정 단계 : 통신이 가능한 모든 노드들을 탐지하여 그룹을 형성하고, 이 중 한 노드를 마스터 노드로 결정한다. 마스터 노드 결정방법은 투표(voting) 방법 등 여러 방법에 적용 될 수 있으나, 본 연구에서는 그룹 내의 노드들은 모두 동등 조건이므로, 노드 번호에 따른 순차적 선정 방법을 적용한다.

(3) 그룹 정보 수집 단계 : 모든 클러스터 노드들이 접근할 수 있는 SDB 공간을 활용하여 그룹들 간의 정보를 비교하기 위하여, 그룹 정보를 작성한다. 그룹 정보는 그룹 내의 노드 개수와 외부 네트워크와의 통신 여부로 구성된다. 외부 네트워크 통신 여부는 그 그룹이 서비스 그룹인지 오류 그룹인지를 판별하는 기준으로 사용한다. 외부 네트워크 통신 여부를 점검하기 위해서는 다양한 운영체제 유틸리티들을 사용할 수 있다.

(4) 그룹 정보 판독 및 기록 단계 : 작성된 그룹 정보를 기록하기 위해 공유 디스크 보드 SDB를 접근한다. 이미 다른 그룹이 작성한 정보가 기록되어 있을 수 있으므로, 기록 전에 먼저 판독연산을 수행하여 다른 그룹의 정보를 읽어온다. SDB에서 판독한 다른 그룹의 정보가 자신이 작성한 그룹 정보 보다 좋은 상황이면, 자신의 그룹을 클러스터 내에서 제거(fence out) 시켜야 하므로 공유 디스크 SDB의 기록 작업을 중단한다. 그렇지 않으면 적정 시간 후, 그룹 정보의 판독 및 기록 연산을 재수행한다. 두 그룹 간의 정보비교는 서비스 그룹이 오류 그룹보다 우세한 것으로 판단하며, 동등한 그룹 내에서는 노드의 개수가 많은 그룹이 우세한 것으로 판별된다.

(5) 오류 처리 및 회복 단계 : 판독 연산 재수행 후 자신의 그룹 정보가 공유 디스크 SDB상에 그대로 남아 있는 경우, 자신의 그룹이 분할그룹의 승자인 서비스 그룹으로 결정되고, 나머지 모든 그룹에 대한 오류 처리 작업을 수행한다. 다른 분할그룹들은 서비스 그룹의 정보를 판독하게 되어 모두 클러스터 내에서 제거된다.

```
[1단계 : 오류 탐지]
1. 주기적 점검 메시지를 브로드캐스트 방식으로 전송한다.
2. if (초기 노드 뷰 != 현재 노드 뷰)
        2.1 CRM에 오류 상황을 통지하고, 현재 통신 가능한 노드 리스트를 전
            달한다.
        2.2 통신 가능한 노드들을 그룹화하고, 마스터 노드를 선정한다.

[2단계 : 분할그룹 문제 해결] (마스터 노드 수행 코드)
1. 자신의 그룹 정보(그룹 내 노드 개수 및 네트워크 상태 점검 결과)를 작성한다.
2. 클러스터내의 공유 디스크인 SDB를 open 한다.
3. 공유 디스크 SDB에 작성된 그룹 정보를 기록-판독 한다.
4. for(i=0; i( 재수행 횟수; i++)
    4.1 if ( 자신의 그룹 정보 )= 판독된 그룹 정보)
            sleep(given times);
    4.2 else /* 패자 그룹 */
        4.2.1 그룹 내의 모든 노드들에게 I/O Fence Out 명령 전송
        4.2.2 클러스터 내에서 제거되기 위하여 I/O Fence Out 루틴 호출
        4.2.3 시스템 패닉
5. if (판독한 그룹 정보 == 자신의 그룹 정보)
            서비스 그룹 = 자신의 그룹;

[3단계 : 회복 단계] (서비스 그룹 노드 수행 코드)
1. 서비스 그룹 노드를 제외한 모든 노드들의 기능을 전담하기 위해 오류 노드들
   의 정보 수집
2. 오류 노드들의 기능을 서비스 그룹 노드들에 분담.
3. 현 서비스 그룹 노드들로 구성되는 클러스터 파일 시스템 재개
```

그림 9. 오류 탐지 및 회복 알고리즘 의사코드
Fig. 9 Pseudo Code of Algorithm for Failure Detection and Recovery

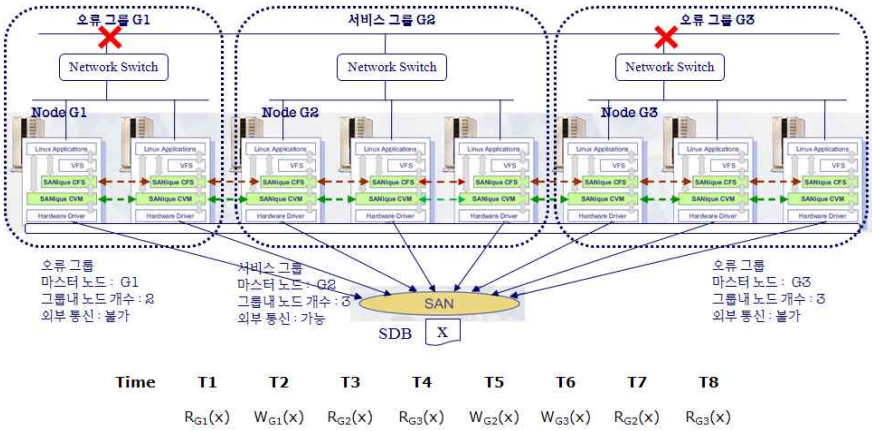

그림 10. 분할된 오류 노드 그룹의 정보 비교 직렬성 위반
Fig. 10. Violation of Serializability for Operation of Split-Brain Node Groups

단계별 회복 방법을 의사코드 알고리즘으로 기술하면 그림 9와 같다. 본 제안된 방법은 분할 그룹으로 나뉘어진 노드들의 구성을 가장 최적화 하여, 최상의 상태의 그룹을 서비스 그룹으로 선정하는 방법으로서 기존의 임의 선정 방식에 의한 열악한 서비스 체제를 방지할 수 있다.

나. 오류 탐지 알고리즘의 정확성 및 직렬성

오류 탐지 및 회복 기법 중에서 그룹 정보 판독 및 기록 단계에서 또 다른 고려사항이 발생한다. 각 그룹들이 통신이 안되는 상태이므로, 공유 디스크 공간 SDB를 접근하는데 있어서 아무런 병행수행 제어가 적용되지 않아, 직렬성에 문제가 발생할 수 있다. 즉 판독연산과 기록연산을 수행하는 중간에 다른 기록 연산이 중복되어 기록될 수 있어, 그룹 정보 비교에서 잘못된 결과를 초래할 수 있다. SDB 내용을 판독하여 테스트하는 동안 다른 그룹이 자신의 정보를 기록하는 직렬성 위반 문제가 발생할 수 있다. 그림 10은 클러스터 노드들이 세 개의 분할그룹으로 나누어 졌을 때, 각 그룹의 마스터 노드들이 자신의 그룹 정보를 기록하고, 다른 그룹의 정보를 판독하여 서비스 그룹을 결정하는 예를 보이고 있다. 그룹 G1과 G3는 네트워크 스위치의 이상으로 외부와 통신이 단절된 오류 그룹이며 그룹 G2는 세 개를 노드를 갖는 서비스 그룹이다.

먼저 시간 T1에서 분할된 오류 그룹 G1이 공유 디스크 공간 SDB에 있는 기록을 판독(RG1(X))하지만, 초기값이므로 자신의 정보를 기록(WG1(X))한다. 그 후 시간 T3에 분할그룹 G2가 G1이 기록한 그룹 정보를 판독(RG2(X))하고, 서비스 그룹 G2가 G1 보다 좋은 조건임을 판단하게 된다. 그러나 G2가 자신의 그룹 정보를 기록하기 전에 시간 T4에 분할그룹 G3가 SDB의 정보를 판독(RG3(X))하고, G3는 G1과 마찬가지로 오류 그룹이지만 노드의 개수가 많으므로 우세하다고 판단하여 자신의 그룹 정보를 기록하려 한다. 시간 T5에 서비스 그룹 G2는 자신의 그룹 정보를 SDB에 기록(WG2(X))하고, 이어서 시간 T6에 오류 그룹 G3가 자신의 정보를 기록(WG3(X))하게 된다. 같은 패턴으로 수차례 재수행 하더라도 분할그룹 G3는 G2보다 열악한 조건이어서 G2의 기록이 남아 있어야만 하지만, 판독-기록 순서상의 직렬성이 위반되어 G3의 그룹 정보가 공유 디스크 SDB에 남게 되고, G3는 시간 T8에 자신의 기록이 남아 있음을 확인하고 결국 자신이 서비스 그룹인 것으로 판별하는 잘못된 결과를 유발하게 된다. 반면에 G2 입장에서 G3의 그룹 정보는 G2 자신의 그룹 정보보다 나쁘므로 자신의 그룹 정보를 지속적으로 남기려고 하지만 최종적으로 자신의 그룹 정보가 아닌 G3의 그룹 정보가 남아있음을 확인하고 스스로 오류 그룹으로 판정하게 된다. 따라서, 오류 그룹 G3가 G1, G2 그룹의 모든 노드들을 클러스터 내에서 제거하고 오류를 회복한 후, 파일 시스템 서비스를 재개하게 되면, 서비스 그룹 G2 보다 제한적인 파일

시스템 서비스를 하게 된다.

회복 관리기 CRM은 분할그룹 정보 비교시 판독후 기록의 연산 방법을 하나의 “test-and-set” 명령어를 사용한다. TS(x) 연산은 R(x) W(x)이 원자적(atomic)으로 수행되는 것으로 정의된다. 따라서, 자신의 분할그룹 정보를 기록할 때, 기존 그룹정보를 비교 판단할 수 없고 단순히 가져오기 연산(fetch)만 가능하게 된다. 먼저 자신의 그룹 정보를 기록 한 후에 판독된 이 그룹정보는 자신의 그룹 정보와 비교에 사용된다. 즉, 판독-판단-기록의 연산 순서가 기록-판독-판단으로 변경되므로, 판단 후에 자신의 그룹 정보가 판독된 그룹 정보보다 우세하다고 판단되는 경우에만, 같은 작업을 재수행한다. 판독된 그룹 정보가 자신의 그룹 정보보다 우세한 경우에는 더 좋은 조건의 분할그룹이 존재한다는 것을 의미하므로 자신의 그룹에 해당하는 모든 노드에게 “fence out” 명령을 보내, 시스템내에서 스스로 제거된다. 재수행 후에 자신의 분할그룹보다 우세한 그룹이 없다고 판단되는 그룹은 서비스 그룹이 되고 나머지 그룹들은 클러스터에서 제거된다. 그림 11은 직렬성 문제를 해결하여 분할그룹 정보를 비교하는 과정을 보이고 있다. 그림 11의 시간 T1에 분할 오류 그룹 G1이 자신의 그룹 정보를 “test-and-set”을 통해 기록 및 판독하고, 시간 T2와 T3에 분할그룹 G2와 G3가 역시 기록-판독 연산을 수행한다. 판독된 정보는 각각의 마스터 노드에서 우열 비교를 판단하게 된다. 이 때, 분할그룹 G3는 G2의 정보를 판독하였으므로 패자로 결정되고, G3 그룹 내의 모든 노드는 최종적인 서비스 그룹에 의해서 제거된다. 그러나 분할그룹 G1과 G2는 각각 그룹 정보 비교시 우세 결정을 하였으므로 시간 T4와 T5에 각각 기록-판단 연산을 재수행한다.

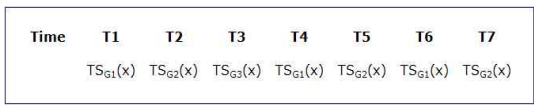

그림 11. Test-and-Set을 이용한 오류 정보 비교
Fig. 11. Comparison of Failure Group Information by Test-and-Set Operation

시간 T4에서 분할그룹 G1은 분할그룹 G3의 정보를 판독하였으므로, 자신의 그룹 정보가 우세하다고 판단하게 된다. 시간 T5에서 분할그룹 G2는 G1의 정보를 판독하여, 자신의 그룹이 우세하다고 역시 판단하게 된다. 그러나 시간 T6에서 분할그룹 G1은 G2의 그룹 정보를 판독하여 자신의 그룹을 패자 그룹으로 결정한다. 한편 시간 T7에서 분할그룹 G2는 최종적으로 자신의 그룹이 최종 서비스 그룹임을 결정하고 자신의 그룹 노드를 제외한 모든 노드들을 오류 처리하게 된다. 따라서 제시된 방법을 통하여 분할그룹들 간에 최적의 그룹이 파일 시스템을 재개할 수 있으며, 또한 동시 접근 제어가 되지 않는 상태에서의 올바른 그룹 정보 비교가 가능하다.

V. 결론

본 논문에서는 SAN 기반의 클러스터 파일 시스템인 SANique™의 시스템 구성도 및 특징에 대하여 기술하였으며, 특히 클러스터 파일 시스템의 분할 오류 노드 그룹에 의해 발생하는 문제점을 조사하였고, 이를 해결하기 위하여 공유 디스크를 활용하는 방법에 대한 기법을 제시하였다. 제안한 기법은 시스템 서비스 온라인 상태에서 수행가능하며, 또한 오류 상황을 정확히 판단하기 힘든 분할그룹 상황에서도 최적의 그룹이 파일 시스템 서비스를 재개할 수 있는 회복 기법을 제시하였다. 클라우드 컴퓨팅과 같이 클러스터 노드 수가 많은 상황에서도 온라인 회복이 가능하도록 상수 시간 내에 오류 그룹을 탐지할 수 있는 방법을 계속 연구중이다.

참고문헌

[1] VERITAS Software Corp., Veritas Volume Manager, http://www.veritas.com

[2] H. Maulshagen, “Logical Volume Manager for Linux”, Sistina Technical Memo, http://www. sistina.com.

[3] MacroImpact, Inc., “SANique Cluster Volume Manager Functional Specification”, MacroImpact Technical Memo, 2008.

[4] 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.

[5] Ghemawat, S., Gobioff, H., and Leung, S. -T. The Google File System, In 19th SOSP, Dec. 2003. pp29-43.

[6] Burrows, M. The Chubby Lock Service for Loosely-Coupled Distributed Systems, In Proc. of the 7th OSDI, 2006. 11

[7] Jeffrey Dean and Sanjay Ghemawat, "MapReduce : Simplified Data Processing on large Clusters", In Proc. of the 5th OSDI, 2004. 11

[8] 김명준 외, 클러스터 기반 통합 멀티미디어 DBMS 개발, 정보통신연구진흥원, 연구결과보고서, 2002. 12.

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

[10] C. C. Fan and J. Bruck, "The Raincore Distributed Session Service for Networking Elements", Proc. Of the International Parallel and Distributed Processing Symposium, 2000.

저자소개

이규웅(Kyu Woong Lee)

1986년 한국외국어대학교 이학사
1990년 서강대학교대학원 공학석사
1998년 서강대학교대학원 공학박사
1998년~2000년 한국전자통신연구원 선임연구원
2000년~ 상지대학교 컴퓨터정보공학부 부교수
※관심분야 : 데이터베이스, 클러스터 시스템 등