2002년 한국정보처리학회 춘계학술발표논문집 제9권 제1호

SAN을 위한 논리 볼륨 관리자 구현

김훈, 이홍석, 최성춘, 윤희용, 추현승
성균관대학교 전기전자 및 컴퓨터 공학부
{hunkim, juspeace, choisc, youn, choo}@ece.skku.ac.kr

Implementation of Logical Volume Manager for Storage Area Network

Hun Kim, Hong Seok Lee, Sung chune Choi, Hee yong Youn, Hyun Seung Choo
School of Information and Communications
Sungkyunkwan University

요 약

현재 인터넷 환경은 다양한 멀티미디어 정보의 폭발적인 증가로 인해 정보의 효율적인 저장 및 관리, 가공을 위한 볼륨 관리자가 요구되고 있다. 따라서 본 논문은 대용량의 데이터 관리를 요구하는 SAN (Storage Area Network)환경에서 대용량 저장 장치 구성을 위한 LDP(Logical Disk Pool)를 설계하고 그 성능을 평가하였다. LDP는 여러 개의 물리적 디스크를 물리적 개념이 아닌 논리적 저장 공간의 개념을 사용하여 관리함으로써, 기존 물리 디스크의 크기 제한과 확장성 문제를 해결하여 SAN 환경에 적합한 대용량 저장장치 구성을 가능하게 한다.

1. 서론

현재의 인터넷은 다양한 멀티미디어 컨텐츠를 제공함으로써 정보는 대용량 고부가가치의 경향을 띄고 있다. 이러한 경향에 맞춰 정보의 효율적인 저장, 관리 그리고 가공을 요구하는 SAN 환경에서 볼륨 관리자의 개발이 절실이 요구되고 있다. SAN 은 폭발적으로 증가하는 데이터의 관리를 위해 호스트 컴퓨터의 종류에 구애 받지 않고 fiber 스위치에 연결된 저장 장치 사이에 대용량 데이터를 전송 시킬 수 있는 새로의 개념의 고속 네트웍 스토리지 시스템이다.

LDP 란 SAN 기반 저장 장치 시스템을 위해 구현된 볼륨 관리자로서, SAN에 연결된 여러 저장 장치를 하나로 묶어 논리적인 가상 저장 장치를 제공 함으로써 기존의 저장 장치 시스템이 가지고 있는 크기 및 성능 제한, 데이터 신뢰성 보장의 한계점을 극복하기 위해 구현된 저장 장치 관리자이다.

기존에는 하나의 물리적 디스크를 파티션으로 나누어 해당 파티션에 파일 시스템을 마운트(mount)하여 사용하였다. 그러므로 파일 시스템의 크기가 물리적 디스크의 용량에 의해 제한되고, 확장이 불가능했다. 하지만 LDP 에서는 물리적 디스크의 개념이 아닌 논리적인 저장 공간의 개념을 사용하여 기존 물리 디스크의 크기 제한을 해결하고, 자유로운 확장성을 제공해준다.

기존의 단일 파일 서버를 통하여 데이터를 전송하는 구조의 예를 들어 보면, 네트웍의 대역폭이 저장장치의 대역폭보다 클 경우, 동시에 요청하는 클라이언트의 데이터 전송요구를 빠르게 서비스 해 줄 수 없다. 따라서 본 논문에서는 Fiber Channel[2]을 이용하여 고속의 데이터 네트워크를 구축한 SAN 환경에서 다중 파일 서버를 사용하여 데이터의 요구를 분산시켜 동시에 다수의 저장장치 대역폭을 활용하는 저장장치 관리를 위한 LDP 를 구현하고 그 성능을 평가 하고자 한다.

본 논문의 구성은 다음과 같다. 2 장에서는 대용량 저장장치에 대한 관련 연구를 알아보고, 3 장에서는 LDP 구현에 대해 설명한다. 4 장에서는 구현된 LDP 모듈의 성능을 평가하고, 마지막으로 5 장에서는 결론을 맺는다.

2. 관련 연구

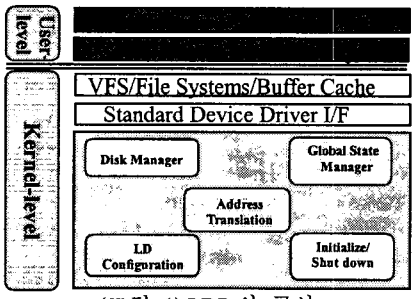
데이터의 효율적인 저장 및 관리, 가공을 위해서 여러 개의 물리적인 저장 장치를 하나로 연결하여 가상의 논리적인 저장 장치로 RAID[5]를 지원하는 볼륨 관리자에 대한 많은 연구가 진행되고 있다. 대표적으로는 Sistina 소프트웨어에서 단일 시스템에 사용될 수 있도록 개발된 Linux LVM[6]이 있다. 하나의 시스템으로 대용량의 정보를 처리하는 것은 현재 폭발적으로 증가하는 네트워크 환경에서는 한계가 있다.

SAN 환경에서는 여러 호스트가 함께 연결되어 동작하므로 동시에 동일 저장 공간을 공유하여 동작하게 된다. 공유되는 저장 공간은 정확한 데이터를 유지하기 위해 동시성 제어가 필요하다. 따라서 기존에 단

일 시스템을 위한 볼륨 관리자는 다중서버를 사용하는 SAN 환경에서 적절한 제어를 수행할 수 없다. 이를 해결하기 위해 최근에 SAN 환경에서 여러 호스트가 저장 장치를 공유할 수 있도록 적절한 제어를 수행하는 볼륨 관리자가 제안되었다. 대표적인 시스템으로는 Linux 에서 동작하는 GFS 의 Pool Driver 가 있다.[1] Pool Driver 는 SCSI 디바이스 드라이버와 파일 시스템 중간에 위치한 커널 내 모듈이며, GFS 파일 시스템과 함께 SAN 환경에서 저장 장치들을 공유하기 위해 사용되어 진다. 그러나 Pool Driver 는 논리 볼륨에 대한 관리 기능이 매우 빈약하다. 실제 물리적인 저장 장치와 가상의 논리적인 저장 장치간에 매핑 구조가 계산 방식을 따르도록 되어 있어, 매핑 관리를 능동적으로 변경하는 것이 불가능함으로써 다양한 관리 기능을 제공하는데 한계를 가진다. 본 논문에서는 이러한 한계를 극복하기 위한 SAN 기반의 LDP 볼륨 관리자를 제안하고자 한다.[3]

3. LDP 구현

LDP 의 구성은 그림 1 에서와 같이, 사용자-레벨(User-level)과 커널-레벨(Kernel-level)로 나뉘어 진다. 사용자-레벨은 실제 사용자에 의해 수행되는 명령어와 그 명령어 들을 위해 필요한 라이브러리들로 구성되어 진다.



(그림 1) LDP 의 구성

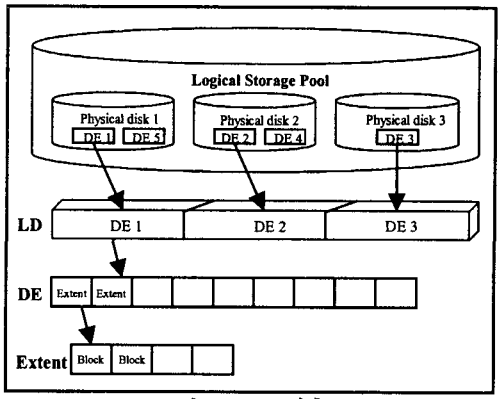
커널-레벨은 사용자-레벨에서 저장 장치와 관련하여 요청된 명령을 처리하기 위해 디스크 관리자, 전역 상태 관리자, 주소 변환 관리자, LD 구성 관리자, 그리고 초기화 관리자의 다섯개 서브 모듈로 나뉘어 진다.

디스크 관리자 모듈은 사용자-레벨을 통해 전달된 디스크 동작 요청을 처리하고, 이에 관련된 통신 메시지를 만들어 다중 서버 환경에서 서버간의 패킷 전송을 통해 전체 서버의 동기화를 맞추기 위한 기능을 수행한다. 전역 상태 관리자 모듈은 각 서버에 의해 공유되는 물리 디스크 정보와 논리 디스크 정보 같은 전역 정보를 관리한다. LDP 에서는 전역 정보의 관리를 위해 마스터 LDP (master LDP)를 사용하여, 변화되는 전역 상태 전송을 위한 책임을 지게 한다.

LD 구성(LD configuration) 모듈은 LDP 를 위해 논리적 디스크를 관리한다. 각 서버들은 LD 구성 모듈과 전역 상태 관리자 모듈에 의해 같은 저장공간을 공유하게 된다.

3.1 LDP 계층구조

LDP 의 계층구조는 그림 2 에서와 같이 논리 저장 공간(Logical Storage Pool), 물리 디스크(Physical Disk), 논리 디스크(LogicalDisk), 드라이브 요소(Drive Element)와 익스텐트(Extent)로 구성된다.



(그림 2) LDP 계층구조

3.1.1 논리 저장 공간

논리 저장 공간은 LDP 의 가장 상위 계층으로 여러 개의 파티션 된 물리 디스크로 구성되며, 물리 디스크의 각 파티션은 LD 를 구성하기 위해 필요한 단일 저장 객체로 동작하게 된다.

3.1.2 논리 디스크(LD)

LD 란 다수 개의 DE 로 구성된 디스크 단위로 사용자가 하나의 파일 시스템을 마운트하여 사용하는 논리 디스크이다. 사용자는 실제로 논리디스크를 하나의 물리 디스크와 같이 인식하고 사용하게 된다.

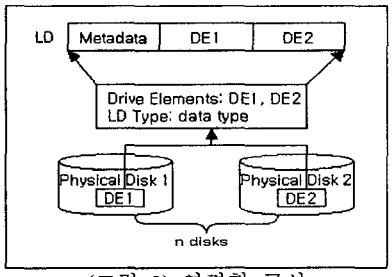
LD 를 구성하는 순서는 시스템에 연결된 모든 디스크를 검색한 후, LDP 에 의해 관리되는 물리 디스크를 찾아 여기에 쓰여진 LD 헤더를 읽어 LD 정보를 만든다. LD 는 64 비트의 주소 공간을 제공함에 따라 기존 디스크의 32 비트 방식 주소 공간으로 발생하는 파일 크기 제한을 확장시켰다.[4] LD 는 DE 의 구성에 따라 하나 또는 그 이상의 물리 디스크로 구성된다.

3.1.3 드라이브 요소(DE)

DE 는 여러 개의 익스텐트들로 구성된 단위이다. 각 익스텐트들은 연속된 블록들로 구성되어 DE 를 구성한다. DE 정보는 각 디스크별로 used 리스트와 unused 리스트를 이용하여 관리하기 위해, 각 디스크의 앞부분에 리스트의 헤더 포인터를 저장하고 있다. LD 를 구성하는 DE 에 대한 정보는 각 LD 의 앞부분에 기록해둔다. 하나의 LD 에는 여러 개의 DE 가 존재할 수 있으며, LD 를 구성하는 DE 의 크기는 모두 동일하게 구성된다. 그 이유는 LD 를 구성할 때, 하나의 LD 를 사용자가 원하는 DE 의 개수로 나누어 구성하기 때문이다. 이렇게 구성하므로서 DE 에 대한 정보를 비트맵을 통하여 용이하게 관리할 수 있다는 장점을 가진다. 반면에 실제 저장 장치의 크기를 고려하

여 DE 를 구성하여야 하기 때문에 만약 크기를 고려하지 않을 경우 작은 디스크 공간은 DE 를 할당 받지 못해 디스크의 낭비를 가져오는 단점이 있다.

DE 를 생성하는 방법에는 여러 가지 방법이 존재하는데, 대표적으로 연결형(concatenation), 미러링(mirroring), 스트라이핑(striping) 방법 등이 있다. 본 논문에서는 DE 를 생성하는 방법으로 연결형 방식을 사용하고 있다. 연결형 방식은 그림 3 과 같이 다수 개의 DE 를 순차적으로 묶어서 하나의 LD 를 구성한다. LD 를 구성하는 DE 들에게는 순서가 정해져 있으며, 그 순서대로 LD 의 논리 주소 공간에 맵핑 된다. DE 추가/제거 시 특별한 제약은 없으며 LD 를 생성할 경우 LD 의 크기와 DE 의 개수가 변수로 입력되어 각 DE 의 크기는 (LD 크기/DE 개수)에 의해 결정된다.



(그림 3) 연결형 구성

3.2 LDP 관리자

3.2.1 구성 관리

구성 관리는 여러 물리 저장 장치를 모아서 가상화한 논리 볼륨을 생성하거나 삭제 또는 변경 등의 작업을 수행하는 역할을 한다. 이를 위해 LDP 의 구성 관리에서는, 주요하게 3 가지 동작을 수행한다.

첫번째는 운영체제에서 제공되는 fdisk 같은 툴을 통해서 생성되는 디스크 파티션(Disk Partition) 또는 물리 디스크(Physical Partition)를 생성한다. 따라서 물리 디스크는 논리 저장 공간의 최상위 구성 단위가 된다. 두번째는 논리 저장 공간이다. 논리 저장 공간은 확장 가능한 물리 파티션들의 집합으로서 이름이 붙여지며 일련의 연속적인 주소 공간을 이루게 된다. 논리 볼륨의 크기는 시스템 운영 중에 변경 가능하며 64 비트 주소 체계를 사용한다. 세 번째는 익스텐트로써 동일 사이즈를 갖는 연속적인 블록의 모임으로 디스크 공간의 최소 단위이며, 각 익스텐트의 크기는 논리 디스크 생성시에 결정된다.

메타 데이터는 전역 데이터와 로컬 데이터로 구분되어 질 수 있다. 전역 데이터는 볼륨 전체에 대한 내용으로 각 디스크 파티션에 중복 저장되게 된다. 로컬 데이터는 해당 디스크 파티션에 하나만 유지하게 된다. 전역 데이터와 로컬 데이터의 구분은 볼륨에 대한 메타 데이터의 변경시 최소한의 변경만으로 성능을 향상 시키기 위한 것이다.

3.2.2 초기화 관리

초기화 관리는 Kernel-level 의 초기화/셧다운(Initialize / Shutdown)모듈에 의해 수행된다. 단일 서버 환경에서 LDP 의 사용은 많은 요구 조건을 필요로 하지 않는다. 단순히 저장 장치의 유무만을 확인하여 동작하게 된다. 하지만 다중 서버 환경에서는 각 서버간의 LDP 정보에 관한 동기화가 필수적이기 때문에, 서버간에 통신을 위해 커뮤니케이션 모듈(communication module)이 필요하게 된다. 커뮤니케이션 모듈이 동작하기 위해서는 각 서버의 환경을 저장한 테이블이 존재하여야 하며, LDP 는 테이블에 저장된 정보를 바탕으로 서버간 통신을 수행하게 된다.

(표 1) 호스트 구성 테이블

Host id	IP address	Timeout	Max hosts
0	123.0.0.1	3	256
1	123.0.0.2	3	256

테이블에 저장되는 정보 중에 LDP 에 관련된 정보는 표 1 에서와 같이 마스터 LDP 번호, 해당 호스트 식별자, 각 호스트의 IP 주소, 제한 시간 등이 저장되게 된다. 실제 LDP 서버간의 통신은 커뮤니케이션 모듈에서 담당하게 되고, LDP 에서는 LDP 에서 수행하는 동작과 동작에 관련된 정보를 패킷으로 만들어 커뮤니케이션 모듈에게 넘겨주는 역할을 담당한다.

초기화 작업은 서버에 연결된 모든 저장 장치를 읽어 구성되지 않은 저장 공간에 저장하게 된다. 만일 저장 공간에 저장된 물리 디스크, LD 와 DE 에 대한 정보가 있다면 해당 정보를 바탕으로 새로이 정보를 수정하게 된다. 이렇게 함으로써 다중 서버간에 동기화를 이루게 된다.

3.2.3 매핑 관리

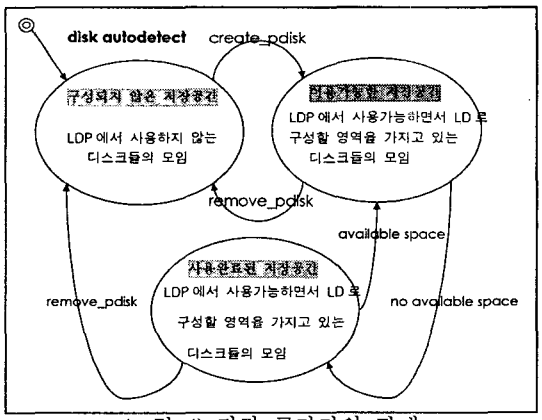
매핑 관리는 익스텐트 단위에 일치하는 물리적 주소를 논리적 주소로 매핑 시켜주는 역할을 담당한다. 일반적으로 파일 관리자나 LDP 를 기반으로 개발될 파일 시스템에서는 모든 I/O 동작을 논리 디스크의 논리 주소 공간에서 수행된다. 그렇기 때문에 이 논리 주소는 실제 물리적인 저장 장치의 물리 주소에 매핑되어야 한다. 물리 주소와 논리 주소를 매핑 시키는 방법에는 계산에 의한 방법과 매핑 테이블을 이용하는 방법이 있다. 계산에 의한 방식은 매핑 테이블을 유지할 필요가 없어서 관리 오버헤드가 없는 장점이 있는 반면에 온라인 재구성이 불가능하다는 단점을 가지고 있다. 그래서 LDP 에서는 온라인 재구성이 가능한 매핑 테이블을 사용하고 있다.

3.2.4 비트맵 관리

LDP 에서는 파일 시스템과 같은 상위 모듈에서 관리하는 논리 주소를 위한 할당 비트맵을 두어 구성하므로 물리 주소의 사용과 논리 관계를 독립시켜 구현할 수 있다. 이것이 익스텐트 할당 비트맵이다. 익스텐트 할당 비트맵은 각 익스텐트 당 한 비트를 사용하여 디스크 파티션 내의 해당 익스텐트의 사용 유무를 표현한다. 이 비트맵은 해당 파티션에 종속적인 정보만을 담고 있으므로 로컬 에러에 대비하여야 한다.

3.2.5 저장공간관리

저장 공간 관리는 시스템에 존재하는 디스크들을 모두 검색하여 물리 디스크로 만들 수 있는 디스크들을 관리한다. 디스크들은 구성되지 않은 저장 공간(unrecognized storage pool), 이용 가능한 저장 공간(available storage pool), 사용 완료된 저장 공간(used-up storage pool)으로 나뉜다. 각 저장 공간(pool)의 특성, 저장 공간에 속하게 되는 경우와 저장 공간을 빠져나가게 되는 경우는 그림 4 와 같다. 초기에 시스템을 시작하면 초기화 모듈에 의해 전체 시스템의 파티션 된 물리 디스크를 탐색하여 모든 디스크들은 구성되지 않은 저장 공간에 속하게 된다. 여기서 사용자 명령어 create_pdisk 명령어를 이용하여 물리 디스크를 생성하게 되면, 해당 디스크는 이용 가능한 저장 공간에 들어가게 되며 생성된 물리 디스크를 삭제하게 되면, 원래의 상태였던 구성되지 않은 저장 공간 상태로 돌아가게 된다. 이용 가능한 저장 공간 상태에서 디스크 공간을 모두 사용하여 더 이상 논리 디스크로 구성할 수 없는 경우에는 사용 완료된 저장 공간의 상태로 가게 되고, 물리 디스크 내부의 LD 또는 DE 가 삭제되어, 사용 가능한 디스크 공간이 발생할 경우 다시 이용 가능한 저장 공간으로 돌아가게 된다. 저장 공간 관리는 위의 상태에 따라 실제 동작들을 제어하게 된다. 또한 저장 공간 정보는 여러 서버들간에 정보를 유지하게 되어, 서버들간에 일관성을 유지하게 된다.

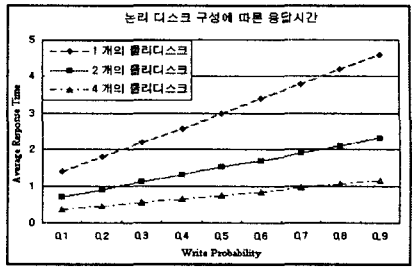


(그림 4) 저장 공간간의 관계

4. 테스트 환경 및 성능분석

1 차 버전은 4 대의 호스트와 2 대의 JBOD 환경에서 구현되었고, 커널 버전 2.2.16 에서 개발 되었다. 호스트와 JBOD 는 광 스위치를 통해 연결되어 동작한다. 현재 본 논문을 위해 구현된 논리 볼륨의 생성 및 삭제, 생성된 볼륨의 정보 변경 등이 가능하며, 지원하는 레이드 레벨은 연결형이다.

그림 5 는 본 논문에서 구현된 논리 볼륨 관리자의 논리 디스크 구성에 따른 성능을 나타낸 것이다. 같은 크기와 동일한 갯수의 DE 를 갖는 논리 디스크를 구성하는데 있어서 다른 갯수의 물리 디스크를 사용한다. 전체 요청 작업은 확률에 의해 읽기와 쓰기로 나뉘어 지고, 다수개의 물리디스크로 논리 디스크가 구성되는 경우에 접근은 순차적으로 이루어 지게 된다.



(그림 5) LD 구성에 따른 응답시간

그림 5 에서 보는 것과 같이 같은 크기의 논리 디스크는 DE 의 갯수와 동일한 물리 디스크를 사용하는 것이 가장 우수한 성능을 나타낸다. 논리 디스크를 구성하는 DE 의 갯수와 물리 디스크의 갯수가 같은 경우 논리 디스크에 요청되는 작업은 동시에 최대 DE 의 갯수만큼 동시 처리가 가능하므로 읽기와 쓰기 모든 면에서 우수한 성능을 나타내게 된다. 논리 디스크를 하나의 물리 디스크로 구성하는 것은 현재 일반적으로 단일 사용자가 논리 디스크를 구성하지 않고 파일 입출력을 사용하는 것과 같은 성능을 나타내게 된다.

5. 결론

본 논문은 폭발적으로 증가하는 멀티미디어 컨텐츠 정보 관리를 위해 호스트 컴퓨터의 종류에 구애 받지 않고, 저장 장치 사이에 데이터를 전송 시킬 수 있는 SAN 환경에서, 대용량 파일 시스템 구성을 위한 LDP 모듈을 설계하고 구현하였다. LDP 는 64bit 주소 공간을 가짐으로써 기존의 32bit 크기 제한을 확장시켰고, 여러 개의 물리적인 디스크를 논리적으로 관리함으로써 폭발적으로 증가하는 데이터를 효율적으로 저장 및 가공하여 정보의 확장성 및 가용성을 제공한다.

참고문헌

[1]David C. Teigland and Heinz Mauelshagen. "Volume Managers in Linux"(Sistina Software, Inc.)

[2]Alan F.Benner. "Fibre Channel: Gigabit Communications and I/O for Computer Network", McGraw-Hill, 1996.

[3]김창수,김경배, 신범주 "SAN 에서의 리눅스 클러스터 볼륨 관리자 개발", 정보처리학회, 제 1 회저장 시스템워크샵

[4]Kenneth W.Preslan et al. "A 64-bit, shared disk file system for linux". The Seventh NASA Goddard Conference on Mass Storage Systems and Technologies in cooperation with the Sixteenth IEEE Symposium on Mass Storage, Pages 22 -41, San Diego, CA, March 1999.

[5]"Raid.edu" – (http://www.acnc.com)

[6]Bert Hubert, Richard Allen "Logical Volume Manager HOWTO"