리눅스 클러스터 파일 시스템을 위한 통신모듈의 설계 및 구현*

박의수° 유찬곤 손호신 최현호 김형식 유관종
충남대학교 컴퓨터과학과
{uspark, jargon, hsson, hyuno, hskim, kjyoo} @cs.cnu.ac.kr

Design and Implementation of The Communication Module for a Linux Cluster File System

Ui-Su Park° Chan-Gon Yoo Ho-Shin Son Hyun-Ho Choi Hyong-Shik Kim Kwan-Jong Yoo
Dept. of Computer Science, ChungNam University

요 약

클러스터 파일 시스템은 기존의 클러스터링 기술을 파일 시스템에 적용하여, 각 노드 단위로 파일 시스템을 구성할 때 발생되는 저장 공간과 대역폭의 제약문제를 극복하기 위한 방법이다. 클러스터 파일 시스템은 하나의 원본 파일을 여러 노드에 나누어 저장하므로, 효율적인 노드간 데이터 통신을 필요로 하며, 노드 내부에서도 클러스터 파일 시스템과 어플리케이션과의 전용 데이터 교환 메커니즘을 지원해야 한다. 본 논문에서는 클러스터 파일 시스템이 안정적이고 효율적인 방법으로 멀티미디어 데이터를 분산 저장하기 위하여 필요한 통신 모듈을 설계 및 구현한다.

1. 서 론

수많은 클라이언트에게 더욱 다양한 멀티미디어 서비스를 제공하기 위해서는 프로세서의 고성능화 및 병렬화가 필요하다[1]. 이는 과도한 비용을 발생시키는 것은 물론 실제 운영시에 가격 대 성능에 한계점을 가진다. 이와 같은 문제점들을 해결할 수 있는 새로운 형태의 서버 모델이 필요하게 되었고 이런 환경 하에서 운영될 수 있는 파일 시스템이 바로 리눅스 기반 클러스터 파일 시스템이다[2]. 이 클러스터 파일 시스템은 데이터 입출력 대역폭을 극대화하여 효율성을 높이고 각 노드의 입출력 부담을 균등하게 부과하기 위하여 원본 파일을 여러 노드에 분산 저장한다. 이렇게 파일을 노드들에 분산 저장하기 위해서는 클러스터 파일 시스템을 구성하는 각 노드의 데몬은 다른 노드에서 실행되고 있는 데몬과 자료를 주고받을 수 있어야 한다.

이러한 일을 가능하게 하는 데몬이 필요로 하는 통신 서비스를 제공하기 위해 통신 모듈을 설계 및 구현한다. 본 시스템은 인터넷 상에서 수많은 클라이언트의 요구에 대해 원활한 처리를 보장할 수 있어야하고, 안정적인 서비스를 제공하여 신뢰성과 경제성 등을 보장하여야 한다[3].

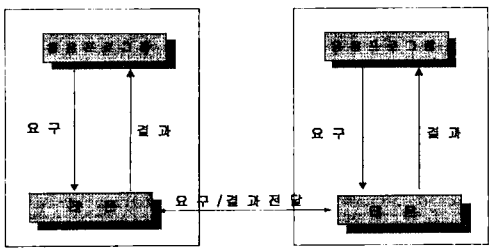
본 논문에서는 리눅스 클러스터의 핵심이 되는 클러스터 파일 시스템의 노드 내 또는 노드간의 원활한 통신을 보장하기 위해 통신 모듈의 구조를 정의하고 모듈간의 데이터 교환 과정을 설계한다. 그리고 노드 내부의 통신을 가능하게 할 Intra-node communication module과 노드와 노드 사이의 데이터 교환을 통한 통신을 가능하게 하는 Inter-node communication module, 즉 통신 모듈 라이브러리를 개발한다.

본 논문의 구성은 다음과 같다. 2장에서 통신 모듈을 정의하며 요구사항을 살펴보고, 3장에서는 통신 모듈을 설계 및 구현하며, 4장에서는 구현환경과 구현된 함수를 살펴본다. 그리고 마지막으로 5장에서는 구현에 대한 결과를 정리한다.

2. 통신 모듈 정의 및 요구사항분석

2.1 정 의

클러스터 파일 시스템을 구성하는 각 노드에는 데몬이 있다. 데몬은 파일 혹은 디렉토리를 관리하고 파일을 다중노드에 분산저장하며 응용프로그램 혹은 노드간의 통신 서비스를 제공하기 위해 구현된 프로그램이다. 통신 모듈은 이러한 일을 하는 데몬에게 필요한 통신 서비스를 제공한다.



[그림 1] 통신모듈의 동작

* 본 논문은 정보통신부 지원 선도기반기술개발사업에 의하여 수행된 과제의 결과임

2.2 통신 라이브러리의 요구사항

통신 라이브러리는 응용프로그램과 데몬, 데몬과 데몬 사이의 통신을 원활히 지원하기 위하여 다음의 데몬 개발자 수준 API 함수를 지원한다.

- 응용프로그램이 필요로 하는 서비스를 데몬에 요청하는 함수
- 데몬이 응용프로그램의 서비스 요청을 처리 후 처리 결과를 돌려주는 함수
- 데몬이 응용프로그램으로부터 받은 서비스 요청을 직접 처리할 수 없어 다른 노드에 있는 데몬에게 서비스 요청을 전달하는 함수
- 데몬이 서비스 요청을 처리한 후 다른 노드의 데몬에게 처리 결과를 돌려주는 함수
- 데몬이 다른 노드의 데몬으로부터 서비스 요청 혹은 결과를 받는 함수

3. 설계 및 구현

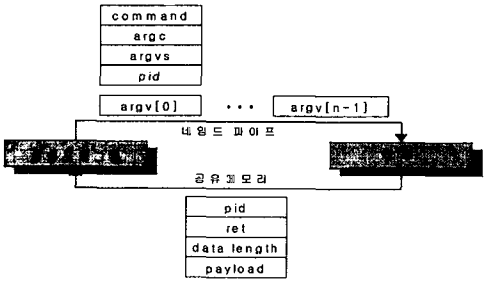
3.1 설계 고려 사항

클러스터 파일 시스템의 통신모듈은 데몬간, 그리고 데몬과 응용프로그램과의 원활한 데이터 교환을 지원하기 위하여 다음과 같은 사항들이 고려되어야 한다.

- 멀티미디어 데이터 등과 같은 대규모 데이터의 처리 성능을 극대화하며 텍스트 등의 작은 데이터 처리성능 향상은 고려하지 않는다.
- Intra-node communication module은 노드 내에서만 운영체제 프로세스 사이의 데이터 교환을 담당하고 교환되는 데이터가 대용량의 멀티미디어 데이터이므로 리눅스상의 가장 속도가 빠른 방법을 사용해야 한다.
- 서비스 요청 및 수락, 처리 결과 대기 시 바쁜 대기가 발생하지 않아야 한다.
- 데드락이 발생하지 않아야 한다.
- Inter-node communication module은 네트워크 초기화 및 작동 시에 다른 노드 데몬 실행 여부에 영향을 받지 않아야 한다.

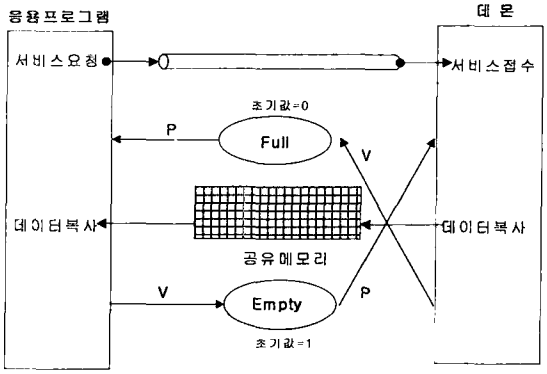
3.2 Intra-node communication library

Intra-node communication library는 동일 노드 내의 응용프로그램과 데몬 사이에서 데이터를 주고받는다. 이때 데몬이 응용프로그램들의 서비스 요청을 받는 방법으로 리눅스 상의 네임드 파이프를 이용한다. 반면에 응용프로그램의 요청에 의해 데몬이 데이터를 제공하는 경우에는 공유 메모리를 이용한다[2]. 이 공유메모리는 동기화가 필요하고 동기화를 위해서 세마포어를 사용한다[4]. [그림 2]에서 볼 수 있는 바와 같이 어플리케이션의 요청을 동일 노드의 데몬에서 바로 처리할 수 있는 경우에는 다른 노드의 데몬과 커뮤니케이션하지 않고 바로 응용프로그램으로 요청한 서비스의 결과를 돌려준다.



[그림 2] 동일 노드 내에서 응용프로그램과 데몬의 패킷교환

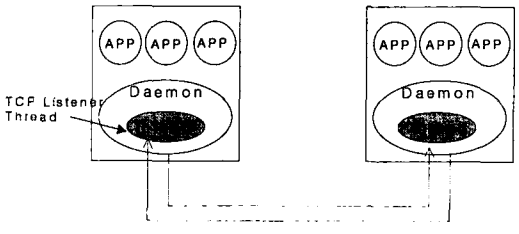
[그림 3]은 응용프로그램과 데몬이 공유 메모리를 사이에 두고 동기화 하는 모습을 보여준다. 예를 들어, 응용프로그램이 데몬에게 클러스터 파일 시스템 상에 존재하는 파일을 읽기 요청을 보내면 데몬은 응용프로그램의 읽기 요청을 받아서 해당 데이터를 공유 메모리에 복사한다. 그리고 또 다른 응용프로그램의 읽기 요청을 받아서 다시 공유 메모리에 데이터를 복사하려할 때, 그 이전의 응용프로그램이 아직 데이터를 읽어 가지 않았다면 공유 메모리 위에 덮어쓰지 않아야 한다. 그러기 위해서 이전 응용프로그램이 데이터를 읽어 가지 않으면 데몬은 'Empty' 세마포어의 P operation에서 계속 대기한다. 그리고 응용프로그램이 공유 메모리로부터 데이터를 읽어 간 후 'Empty' 세마포어에 대해서 V operation을 하면 비로소 데몬은 다시 공유 메모리로 해당 데이터를 복사하고 요청한 응용프로그램이 데이터를 읽어 갈 수 있도록 한다. 'Full' 세마포어는 각 응용프로그램이 생성 및 종료 될 때 동시에 같이 생성 및 종료된다.



[그림 3] 응용프로그램과 데몬의 공유 메모리 동기화 방법

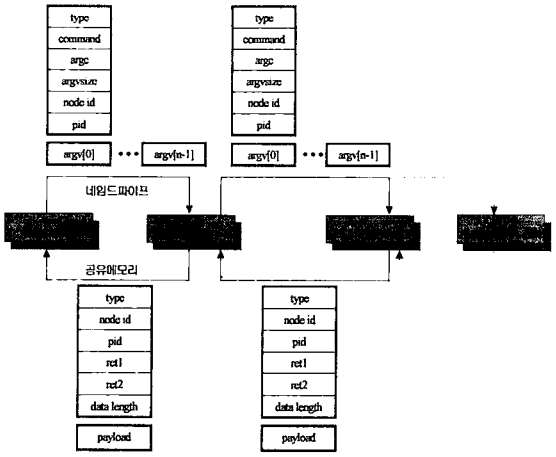
3.3 Inter-node communication library

Inter-node communication library는 여러 노드에 존재하는 데몬 사이의 통신을 수행한다. 노드 간 통신은 기본적으로 TCP/IP를 사용하며 클러스터 파일 시스템 데몬을 실행하면 데몬 초기화 단계에서 다른 노드의 데몬과 TCP 연결을 설정한다. 이미 실행 중인 데몬은 새로운 데몬의 TCP 연결 요청을 받고, 연결된 데몬과 필요 시 데이터 교환을 한다. 데몬은 별도의 TCP Listener Thread를 두어 응용프로그램이나 다른 노드의 서비스 요청을 처리함과 동시에 다른 노드의 TCP 연결 요청을 수락 할 수 있다[그림 4].



[그림 4] TCP Listener Thread

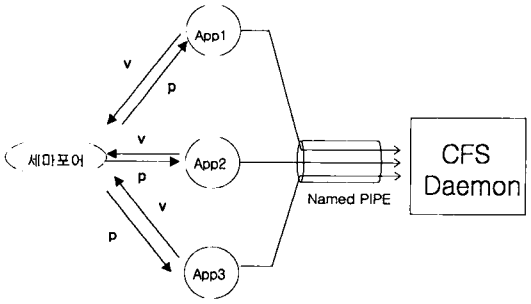
Inter-node communication library는 노드 간 데이터 교환을 위해서 [그림 5]와 같은 패킷이 교환된다.



[그림 5] 노드 간 패킷 교환

3.4 다중클라이언트 지원

클러스터 파일 시스템은 한 노드에서 다수의 응용프로그램에게 클러스터 파일 시스템 서비스를 제공하며, 응용프로그램들의 서비스 요청을 받는 방법으로 리눅스 상의 네임드 파이프를 이용한다[2]. 그런데 네임드 파이프 자체가 I/O atomicity를 보장하지 않으므로 다수의 응용프로그램이 서비스를 요청하면 여러 응용프로그램의 요청 패킷이 섞이는 결과를 초래한다. 그래서 각 응용프로그램이 데몬과 데이터 교환을 위해서 사용하는 Intra-node communication module은 [그림 6]과 같이 App1이 패킷을 모두 보내지 않은 상황에서 App2로 문맥교환이 발생해도, App2는 App1이 V operation을 하기 전까지 데이터를 네임드 파이프에 쓸 수 없다. 이 방법을 통해 패킷이 혼재하는 것을 막는다.



[그림 6] 세마포어를 이용한 다중클라이언트 지원

4. 구 현

본 통신 모듈이 구현된 환경은 [표 1]과 같다.

운영체제	리눅스 커널 2.2.x 혹은 2.4.x
컴파일러	g++, Make 유틸리티
네트워크	TCP/IP over Ethernet

[표 1] 구현환경

통신모듈이 제공하는 API 함수들의 리스트는 [표 2]와 같다.[5]

응용프로그램과 데몬간의 통신을 위한 함수	_send_request	요청전달 함수
	_receive_response	데몬으로부터 응답을 받는 함수
노드와 노드간의 통신을 위한 함수	_receive_message	응용프로그램이 다른 노드의 데몬으로부터 요청 혹은 응답을 받는 함수
	_forward_request	요청을 다른 노드의 데몬에게 보내는 함수
	_send_response	응답을 동일 노드의 응용프로그램에게 보내는 함수
	_forward_response	응답을 다른 노드의 데몬에게 보내는 함수

[표 2] 통신모듈이 제공하는 함수들

5. 결 론

본 논문에서는 클러스터 파일 시스템의 통신 모듈에 요구되는 기능을 규명하고, 그 기능을 효과적으로 구현하는 방안을 제시한 후에, 클러스터 파일 시스템이 필요로 하는 통신 모듈을 설계 및 구현하였다. 본 통신모듈은 데몬과 응용프로그램간, 노드와 노드사이의 멀티미디어 데이터 송수신 기능을 효과적으로 수행한다. 데몬과 응용프로그램간의 통신을 위하여 네임드 파이프로 하여금 응용프로그램으로부터 데몬으로의 패킷을 전송하도록 하였고 다수의 응용프로그램이 데몬에 서비스를 요청하기 때문에 발생될 수 있는 패킷 섞임 현상을 세마포어를 이용하여 해결하였다. 데몬에서 응용프로그램으로의 패킷전송을 위해서는 공유메모리를 사용하였고 노드와 노드간의 통신을 위해 TCP/IP를 사용하였다. 향후 연구과제로는 성능평가를 통하여 처리율을 개선하고 지연시간을 줄이기 위한 최적화가 필요하다.

6. 참고문헌

[1] Stephen E. Deering and David R. Cheriton, "Multicast Routing in Datagram Internetworks and Extended LANs", ACM Transactions on Computer Systems, Vol. 8, No. 2, May. 1990

[2] Rajkumar Buyya, *High Performance Cluster Computing Programming and applications*, Volume 2, 1999

[3] H. Schulzrinne, A. Rao, R. Lanphier, "Real Time Streaming Protocol(RTSP)", RFC 2326, April. 1998

[4] W. Richard Stevens, *Advanced Programming in the Unix Environment*, November. 1998.

[5] 강미연, 홍재연, 김형식, "효율적인 멀티미디어 서비스를 위한 리눅스 클러스터 파일 시스템", 한국정보과학회, 2002 봄 학술발표 논문집.