Windows NT 커널에서 Disk Mirroring 기능 구현

o
김성관*, 장승주*, 지동해**, 김학영**, 이정배***
{niceguy, sjjang}@hyomin.dongeui.ac.kr
동의대학교 컴퓨터공학과*
한국전자통신연구원 프로그래밍 환경 연구팀**
부산외국어대학교 컴퓨터공학과***

Implementation of Disk Mirroring Function in Windows NT Kernel

Kim Sung-Kwan*, Jang Seung-Ju*, Gi Dong-Hea**, Kim Hak-Young**, Lee Jung-Bea***
Dongeui Univ., Dept. of Computer Engineering*
ETRI, Programming Language Section**
PUF, Dept. of Computer Engineering***

요 약

최근 PC 의 고성능화로 PC 기반의 서버 사용이 증대되고 있다. 특히 Windows NT 는 인터넷 서비스를 제공하는 PC 기반 서버로 폭 넓게 사용되고 있다. 이러한 서버에서의 데이터 파괴는 막대한 손실을 가져 올 것이다. 데이터의 안정성과 고 가용성을 위한 방법으로 disk mirroring 기법이 여러 분야에서 사용되고 있다. 기존의 연구들은 UNIX 플랫폼에 편중되어 있고, 현실적으로 사용이 증대되고 있는 Windows 에 대한 연구는 상대적으로 빈약한 상태이다. 본 논문에서는 Windows NT device driver level 에서 다수의 node 에 대한 disk mirroring 기능 구현을 설계한다. Windows NT 는 계층화된 driver layer 로 구성되어 있으며, 구현된 mirroring module 을 드라이버 계층상에 추가함으로써 기존의 기능을 변경하지않고 새로운 기능을 추가할 수 있다.

1. 서론

최근 PC 의 성능이 워크스테이션에 뒤지지 않을 만큼 크게 발전하고 저장매체 또한 고용량, 고성능화 되고있다. 이러한 PC 를 사용한 서버의 사용이 증대되고 있고, 사용하는 운영체제로 기존 UNIX 에서 Windows NT 의 사용이 급증하고 있다.

또한 인터넷의 급속적인 확산으로 네트웍을 통한 고용량의 데이터를 전송할 수 있는 기술이 개발되고 가정에까지 널리 보급되고있다. 이 순간에도 수 많은 사용자가 인터넷 서비스를 제공하는 서버에 접속하고 있을 것이다.

이러한 수 많은 사용자가 사용하는 서버에 저장된 데이터가 파괴될 경우 경제적 손실은 물론 서버의 동작이 비정상적으로 수행되는 등의 치명적인 피해를 입게 될 것이다. 이러한 저장 매체의 데이터를 보호하기 위한 기법으로 여러 가지 방안들이 연구되고 있는데, 대표적인 방법으로 RAID(Redundant Arrays of Inexpensive Disks) 기법이 사용되고 있다[1]. RAID 시스템은 신뢰도를 높이기 위해 패리티 등의 추가정보를 사용하며, 배치에 따라 level0 에서 level6 까지의 방법이 많이 사용되고있다.

이중에서 disk mirroring 기법은 RAID level1 에 해당하며 물리적인 disk 를 재배치하여 동일한 데이터를 중복 저장함으로 해서 높은 신뢰도를 제공한다[2]. 본 논문에서는 네트웍을 통하여 연결된 여러 node 의 disk 를 mirroring 할 수 있는 기능을 Windows NT kernel level 에서 지원할 수 있도록 설계, 구현한다. 본 논문의 구성은 2 장에서 관련연구 사항을 언급하고, 3 장에서 구현 시스템을 설명한다. 그리고 4 장에서는 개선사항과 결론을 맺는다.

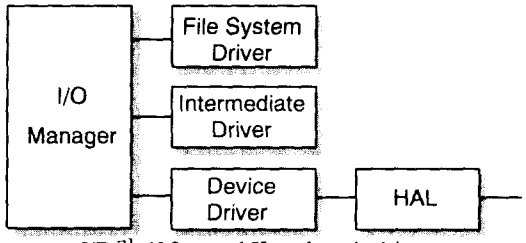
2. 관련연구

2.1 Windows NT Driver Layer

Windows NT 는 Micro Kernel 을 기반으로 하고 있으며, system service 는 각 기능을 담당하는 관리자 모듈이 수행하고 하드웨어에 대한 처리는 HAL(Hardware Abstrac-

tion Layer)을 통해서 이루어진다[3, 5, 6].

Windows NT 의 커널부분은 잘 정의된 계층구조를 가지고 있으며, 커널 모드에서 기능을 수행하기 위해서는 kernel mode driver 형태로 계층구조에 포함시켜야 한다.



[그림 1] Layered Kernel-mode drivers

Windows NT 에서 전체 시스템의 I/O 를 관리하는 I/O Manager 는 커널 모드 드라이버에 대한 framework 을 제공한다[3,4]. 응용프로그램이나 운영체제의 다른 부분에서 발생하는 모든 I/O 요청은 I/O Manager 를 통해서 드라이버에 전달된다.

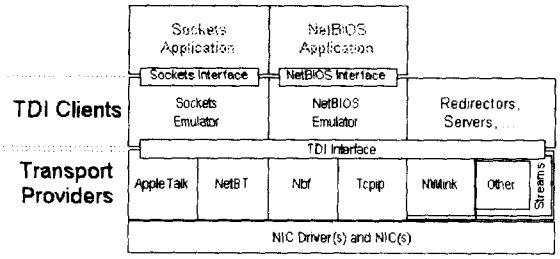
2.2 Kernel-mode Driver 의 기능

2.2.1 Filter Driver

Windows NT 의 드라이버 계층에는 여러 가지 형태의 드라이버들이 존재한다. 그 중에서 filter driver 는 특수한 형태의 드라이버로 기존에 존재하는 드라이버 계층 사이에 추가되어 기존의 코드를 변경하기 않고 특정한 드라이버로 전달되는 요청을 가로채어 새로운 기능을 수행할 수 있는 드라이버이다[4, 5, 6].

Disk mirroring 과 같은 기능에서는 디스크상의 변경된 내용을 파악하고 있어야 하므로 disk driver 나 file-system driver 에 attach 되는 형태의 filter driver 로 구성되어야 한다.

2.2.2 TDI Driver



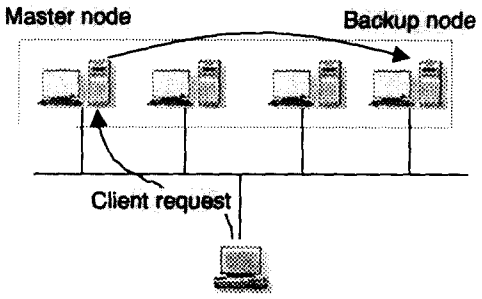
[그림 2] Transport Driver Interface

Kernel mode driver 에서도 다양한 protocol 을 사용하여 원격지의 시스템과 통신을 할 수 있다. Windows system 에서는 TDI(Transport Driver Interface) 계층을 제공하는데, 이러한 interface 를 사용하면 전송 protocol 에 대한 상세한 내용을 알 필요가 없이 이용 가능한 전송 매체와 통신이 가능하다[8].

3. Mirroring system 구현

3.1 시스템 구성 및 기능

Mirroring pair 를 구성하는 시스템은 클라이언트에게 서비스를 제공하는 master node 와 디스크의 내용을 mirror 하는 backup node 로 구성된다. 본 논문에서 구현한 시스템은 master node 의 특정한 드라이브에 대한 데이터가 변경될 경우 이 리스트를 log 파일에 기록하고, 주기적으로 log 파일에 기록된 리스트에 데이터를 backup node 로 전송하게 된다. Backup node 에서는 수신되는 데이터와 관련 정보를 이용하여 master node 의 디스크 배치와 동일한 내용으로 저장하게 된다.



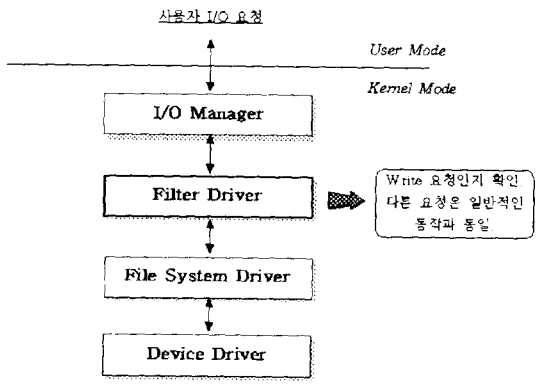
[그림 3] Mirroring 시스템 구성도

File-system filter driver 를 통해서 해당 드라이브로 전달되는 모든 I/O 요청을 모니터링한다. 이 요청 중에서 데이터의 내용을 변경하는 write 요청이 들어올 경우 이 요청의 대상이 되는 파일의 경로명을 log 파일에 기록하게 된다. Master node 의 TDI 전송 driver 는 주기적으로 log 파일의 내용을 읽어 들여 backup node 로 데이터를 전송하게 된다. 이때 실제 데이터를 전송하기 전에 backup node 가 alive 상태인지를 확인하기위한 메시지를 전달한다. 만약 주어진 시간 내에 응답 메시지가 도착하지 않을 경우 backup node 에 이상이 발생한 것으로 간주하고 전송을 연기한다. Backup node 의 수신 TDI driver 는 수신되는 데이터를 master node 와 동일한 경로상에 저장하게 된다.

3.2 Filter Driver 구현

Windows NT 에서 발생하는 모든 I/O 요청들은 내부적으로 IRP(I/O Request Packet)라는 데이터 구조로 전달된다. IRP 는 I/O Manager 에 의해서 생성되어 해당하는 driver 로 전달된다. 이 데이터 구조의 필드 중에서 MajorFunction 과 MinorFunction 필드에는 어떠한 종류의

operation 인지를 명시하는 내용이 들어있다[4, 7].



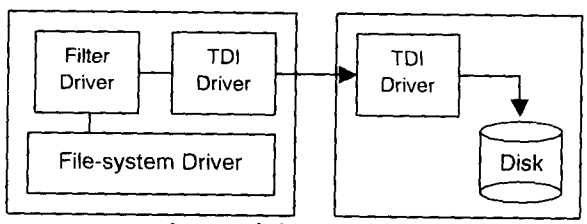
[그림 4] Filter driver 의 기능

개발한 filter driver 는 file-system driver 의 상위에 attach 된다. driver 가 load 된 후에는 file-system 으로 전달되는 모든 I/O 요청을 모니터링 할 수 있다. File-system 으로 전달되는 I/O 요청 중에서 쓰기 요청(IRP_MJ_WRITE)을 가로채어 file-system 에서 모든 동작을 완료한 후 IRP 의 정보를 얻어 요청의 대상이 되는 파일을 log 파일에 기록하고, 이외의 다른 요청일 경우에는 일반적인 동작과 동일하게 동작하도록 전달한다.

3.3 TDI Driver 구현

본 논문에서 구현한 시스템의 네트웍기능은 TDI 계층을 통한 UDP protocol 을 사용한다. 전송 driver 에는 상대 node 의 alive 상태를 파악하기 위한 heartbeat 체크 기능과 log 파일을 읽어 해당하는 파일을 전송하는 기능을 가지고 있다. 수신 driver 는 체크 메시지가 들어왔을 때 응답하는 기능과 수신된 데이터를 local disk 에 저장하는 기능을 가진다.

전송 TDI driver 는 주기적으로 동작하게 되며 변경된 내용이 없거나 backup node 가 실행중이 아니면 전송을 연기한다. 수신 TDI driver 는 system thread 로 동작하며 데이터 수신에 대해서 항상 대기하게 된다.



[그림 5] 구현된 driver 의 구성

4. 결론 및 개선사항

Disk mirroring 기법은 disk 를 중복 배치함으로 해서 추가적인 하드웨어 비용이 요구되기는 하지만, 데이터의 높은 신뢰도를 제공할 수 있어 많은 시스템에서 사용되고 있다.

현재 Windows NT 에서도 disk mirroring 기법을 지원하지만 동일한 disk 매체와 controller 를 요구한다[9]. 본 논문에서 제시하는 시스템 모델은 네트웍을 통하여 다수의 node 를 backup node 로 mirroring 할 수 있도록 설계되어 있다.

현재 구현된 시스템은 1 to 1 node 방식으로 되어있으며, 다수의 node 를 지원할 수 있도록 N to 1 방식으로 구현중이다. 또한 각 서버의 상태를 모니터링할 수 있고, 전체 disk 혹은 일부 disk 만을 mirroring 하도록 설정할 수 있는 configuration 응용프로그램과 통합 작업이 이루어지고 있다.

향후 현 시스템에서 고려되어야 할 사항으로는 서비스를 제공하는 서버에 장애가 발생할 경우 사용자에게 지속적인 서비스를 제공할 수 있는 고 가용성의 기능을 제공하는 것이다. backup node 에 동일한 데이터들이 저장되어 있으므로 장애가 발생할 경우 사용자의 접속이 장애가 발생한 서버에서 backup node 로 변경된다면 데이터 안전성과 고 가용성을 가진 시스템을 구성할 수 있을 것이다.

참고문헌

[1] Shenze Chen, don Towsly, "Performance of a Mirrored disk in a Real-Time Transaction System", ACM SIGMET – RICSM Performance Evaluation, Vol5, 198-207, 1991

[2] P. M. Chen et al, RAID: High-Performance, Reliable Secondary Storage. ACM Computing Survey, vol26, no.2, June 1994, pp.145-185

[3] David A. Solomon, Inside Windows NT Second Edition. Microsoft Press, 1998.

[4] Rajeev Nagar, Windows NT File System Internals, O'Reilly, 1997.

[5] Art Baker, The Windows NT Device Driver Book, Prentice Hall, 1997.

[6] Peter G. Viscarola, W. Anthony Mason, Windows NT Device Driver Development, Macmillan, 1999.

[7] Walter Oney, Programming the Windows Driver Model Microsoft Press, 1999

[8] Windows NT DDK Documentation.

[9] Microsoft High Availability http://www.microsoft.com/technet/avail/default.htm