DVR시스템을 위한 저널링 파일시스템의 성능평가

권우일[0] 윤미현 이동준 장재혁 양승민
숭실대학교 컴퓨터학과
(sunny[0], miya, cometodj, jhjang)@realtime.ssu.ac.kr, yang@computing.ssu.ac.kr

Performance Test of Journaling File System for DVR System

Woo-il Kwon[0] Mi-Hyun Youn Dong-Jun Lee Jae-Hyuk Jang Seung-Min Yang
Dept. of Computer, Soong-Sil University

요 약

최근 무인 감시 시스템 분야에서 리눅스 기반 DVR(Digital Video Recording)시스템의 사용이 늘어나고 있다. DVR시스템은 기존의 아날로그 비디오영상 기록을 대체하여, 카메라에서 입력 받은 아날로그 신호를 디지털화(MPEG또는 MJPEG)된 영상을 하드디스크에 저장하는 방법으로 작동하는 주기적으로 빈번한 하드 디스크의 쓰기 작업을 수행한다. 또 기록되는 내용의 특성상 높은 신뢰도가 요구된다. 따라서 저장장치의 신뢰도가 중요하며, 만일의 사태에 발생할 수 있는 시스템 고장에도 저장된 정보의 손실을 최소화해야 한다. 저널링 파일시스템은 이러한 요구사항에 적합한 기능을 갖추고 있으며 최근 사용범위가 넓어 지고 있다. 본 논문에서는 리눅스 시스템에서 널리 사용되는 4가지 저널링 파일시스템의 상대적인 읽기, 쓰기 및 복구 성능을 비교평가하고 이를 바탕으로 고 신뢰도의 DVR시스템에 요구되는 적절한 저널링 파일시스템을 제안한다.

1. 서 론

리눅스가 발전해 가면서 그 활용분야도 넓어지고 있다. PDA와 같은 소형 내장형 시스템에서부터 네트워크 라우터, 클러스터링 시스템에 이르기 까지 활용범위는 더욱 다양해지는 추세이다. 최근에는 무인 감시 시스템 분야에서 기존의 비디오 테이프를 이용한 아날로그 비디오 영상 저장장치를 대체하는 DVR(Digital Video Recording)시스템에 리눅스의 사용이 늘어나고 있다. DVR시스템은 카메라에서 들어오는 영상신호를 디지털화된 형식(MPEG또는 MJPEG)으로 변환하여 그 내용을 디스크에 압축 저장함으로써 고품질의 영상저장과 빠른 검색을 가능하게 한다[1]. 또 시스템을 네트워크를 이용해서 원격으로 제어할 수 있다는 장점이 있어 점차 그 이용분야가 증가하고 있다. 특히 DVR시스템의 특성상 저장된 내용의 중요성과 여러 대의 카메라에서 입력되는 비디오 스트림의 빈번한 쓰기작업 때문에 디스크 파일시스템의 신뢰성이 특히 우선적으로 고려되어야 한다.

일반적인 운영체제에서 사용되는 파일시스템은 입출력 속도를 높이기 위해 미리 정해진 주 메모리에 캐시의 목적으로 버퍼를 두어 사용한다. 하지만, 버퍼의 내용을 디스크에 쓰기 전에 시스템의 갑작스런 고장이나 정전으로 인하여 재 부팅될 경우, 버퍼에는 삭제된 내용이 디스크에 남아 있게 되어 일관성문제가 발생하여 시스템에 심각한 손상을 초래할 수 있다[2]. 일반적인 파일시스템은 손상된 내용의 복구에 대한 기능이 미약하고 상당히 오랜 복구시간을 필요로 한다. 최근 몇몇 유닉스 시스템과 리눅스 시스템에 적용되고 있는 저널링 파일시스템[3]은 데이터 베이스에서 사용되는 데이터 복구방법을 이용하여 파일시스템의 복구능력을 향상시킨 것으로 앞에서 열거한 DVR시스템의 특성에 적합한 파일시스템으로 EXT3[4], ReiserFS[5], XFS[6], JFS[7]등이 널리 사용되고 있다.

본 논문의 2절에서는 DVR시스템에 대한 설명과 요구되는 파일시스템의 특성을 분석하고, 3절에서는 저널링 파일시스템의 종류와 각각의 특징을 설명한다. 4절에서는 여러 가지 파일시스템에 대한 읽기/쓰기 성능을 실험을 통한 결과를 제시하고, 마지막으로 5절은 결론으로, 실험결과를 바탕으로 DVR시스템에 가장 적합한 파일시스템을 제안한다.

2. DVR시스템에서 요구되는 파일시스템

DVR시스템은 여러 대의 카메라에서 입력된 아날로그 신호를 디지털영상으로 인코딩한 후 디스크에 저장하는 방식으로 동작한다. 디지털 영상으로 인코딩된 내용은 시스템에서 설정된 속도로, 영상의 품질에 따라 대략 10~60KB 크기의 파일을 계속해서 오랜 시간 동안 하드디스크에 기록하기 때문에 하드디스크와 파일시스템의 신뢰성을 요구한다. 또 시스템 용도의 특성상 기록된 정보의 내용이 중요하기 때문에 오류발생시 복구능력이나, 시스템 고장상황에서도 자료손실을 최소화 하는 능력을 제공해야 한다. 본문에서는 DVR시스템에 요구되는 특성 중 파일시스템에서 요구되는 성능과 신뢰성에 대한 평가에 한정한다.

3. 저널링 파일시스템

본 절에서는 성능평가 대상이 되는 저널링 파일시스템

의 특징에 대해서 기술한다. 여기서는 리눅스 시스템에서 사용되는 저널링 파일시스템 중 EXT3, ReiserFS, XFS, JFS에 대해서만 다루기로 한다.

3.1 EXT3

ext2 파일시스템에 저널링 기능을 추가한 것으로 기본적인 구조는 ext2 파일시스템과 동일하여, ext2에서 ext3로의 즉시 변환(in-place conversion)이 가능하다. 복구를 위해 사용하는 저널 파일(journal file)은 일반 파일(regular file)로 존재하며, 여기에 플래그들과 추가적으로 슈퍼블록(super-block)에 저널파일의 i-node number set이 존재하며, 공유 저장장치가 연결되어 있는 클러스터의 다른 노드(node)의 파일시스템까지도 복구할 수 있다.

3.2 ReiserFS

ReiserFS는 플러그인 기반의 객체지향 형태로 구성되어 있다. Balanced tree알고리즘에 기반을 두어 수행성능에 있어 강건한(robust) 장점이 있다. Balanced trees는 블록 할당 기반의 파일 시스템보다 더 공간 효율적이다. ReiserFS 파일시스템의 핵심부분은 B*Tree (B+ Tree의 발전된 형태)에 기초하여 모든 파일시스템의 객체가 하나의 B*Tree 안에 존재한다. 공간효율적인 면에서 다른 파일시스템은 파일들이 각각의 블록에 저장되는데 비해 많은 파일들을 하나의 블록 안에 저장할 수 있다. 또한 i-node에 할당되는 공간이 고정되어 있지 않아 하드디스크의 공간을 절약할 수 있다. extent는 아직 지원하지 않고 있으나, 여러 항목들을 주 파일시스템인 B*Tree안에서 동적으로 생성하고 배열시키므로 전체적으로 디스크 공간을 절약할 수 있다.

3.3 XFS

XFS는 SGI에서 개발된 파일시스템으로 아이릭스 운영체제 기반의 유닉스에서 사용되던 파일시스템으로 리눅스로 포팅되었다. 리눅스용 XFS는 리눅스 2.4커널과 함께 운영되며 파일시스템의 특징들을 지원하는 도구 세트를 가지고 있고, 관리되고 있는 파일들의 수에 상관없이 신속한 복구가 가능하다. 또 확장성과 다른 리눅스 서브시스템들과의 통합이 뛰어나다. 완전한 64-bit 파일시스템으로서 수백만 테라 바이트 용량의 디스크를 지원할 수 있으며, 거의 raw I/O 수준의 대역 폭을 지원한다.

3.4 JFS

JFS는 트랜잭션지향적이며, 고성능의 시스템을 위한 로그(log) 기반의 파일 시스템으로 Scalable하며, 강건한 구조를 가지고 있다. 주요 특징으로는 extent기반의 어드레싱 구조를 사용하며, 디스크상의 물리주소를 가진 파일 내에서 논리적인 오프셋을 매핑 하기 위해 콤팩트하고 효율적이며 종합적인 블록 할당정책을 가지고 있다. 또 동적인 i-node 할당 방법을 사용하여 사용되지 않는 i-node는 해제하여 디스크 저장공간을 절약할 수 있으며, i-node 기반 구성과 B+ Tree 구조의 디렉터리 구성을 지원한다.

4. 실 험

본 실험은 앞에서 언급한 저널링 파일시스템과 ext2 파일시스템의 읽기, 쓰기 작업 시 소요되는 시간과 복구 능력을 측정하기 위한 성능평가이다. 여기서는 저널링 파일시스템 외에 ext2 파일시스템에 대한 성능평가도 실시한다. ext2 파일시스템은 저널링 기능을 포함하고 있지는 않지만, 현재 리눅스의 기본 파일시스템으로 가장 널리 사용되고 있다. 여기서는 각각의 저널링 파일시스템과의 비교대상으로 동일한 실험을 실시한다.

실험환경은 Intel 펜티엄4 시스템으로 256MB의 메모리와 GNU/debian 리눅스 운영체제를 사용한다.. 리눅스 커널 버전은 2.4.16이며, 각 파일시스템당 10GB의 디스크를 할당한다. 단 모든 파일시스템의 버퍼는 모두 512B로 동일하며 모든 실험은 10회를 실시하여 소요시간을 합산하여 비교한다. 실험 시 시스템 환경에서 발생할 수 있는, 실험과 무관한 다른 프로세스에 의한 시스템 부하가 발생할 수 있으나 동일한 환경에서 이루어진 실험이므로 이는 무시한다. 실험 항목은 다음과 같다.

■ **쓰기 실험**

DVR시스템이 영상신호를 JPEG 이미지로 변환하는 경우파일의 크기는 약 수 십KB정도가 된다. 여기서는 10KB에서 60KB의 크기까지 10KB씩 증가 시켜 가면서 파일을 디스크에 쓰도록 한다. 또 여러 대의 카메라에서 신호가 입력되는 경우를 가정해서 쓰레드의 개수를 1, 4, 16개로 설정하였으며, 응용프로그램의 레코드의 크기는 1024B, 2048B, 4096B로 설정한 후 디스크에 쓰는데 걸리는 시간을 측정한다. 또 영상신호를 MPEG과 같은 동영상으로 변환하는 경우를 가정할 수 있는데, 이때 생성되는 파일의 크기는 수 백MB이상의 크기를 가진다. 본 실험에서는 128MB와 256MB 크기의 파일을 사용하여 실시한다.

■ **읽기 실험**

읽기 실험은 쓰기 실험에서 생성된 파일을 읽는데 소요되는 시간을 측정한다. 쓰레드의 개수, 레코드의 크기는 쓰기 실험과 동일하다.

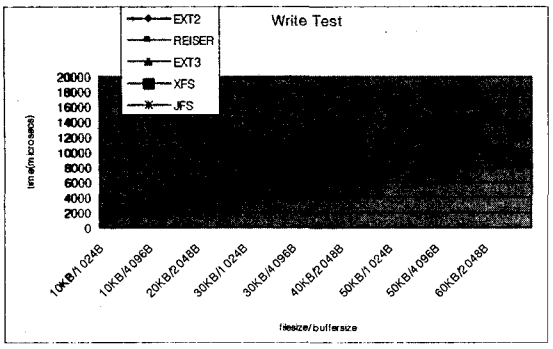
■ **추가쓰기(Append) 실험**

DVR시스템에서 영상신호가 MPEG으로 변환되는 경우 이미 존재하는 파일에 계속적으로 기록해야 하는 경우가 발생한다. 추가쓰기 실험은 쓰기 실험에서 생성된 대용량 파일에 동일한 크기만큼 추가쓰기를 실시한다.

그림 1은 10~60KB크기의 파일에 대한 쓰기, 읽기 실험 결과이다. 그림 2는 512MB와 1024MB크기의 파일에 대한 읽기, 쓰기, 추가쓰기 결과이다.

■ **복구(Recovery) 실험**

파일시스템에 기록되어 있는 파일에 대한 I/O 작업 중에 시스템의 전원을 완전 차단한 후 시스템을 재 시작하여 손상된 파일의 복구 시간과 복구된 정도를 측정하는 실험이다. 그러나 각 파일 시스템에 대한 실험 마다 정확하게 동일한 시간에 전원을 차단하기가 어렵고 따라서 각각의 오류 상황 시 I/O의 진행률이 다르기 때문에 정확한 평가가 어려우나 10회의 동일한 실험 결과, 대략적으로 XFS가 복구율에서 가장 우수한 결과가 나타난다. 또 재 시작 시의 파일시스템 검사 시간은 ext2를 제외한 모든 저널링 파일시스템이 거의 동일하게 무시할 수 있을 정도로 빠른 복구시간을 보여준다.



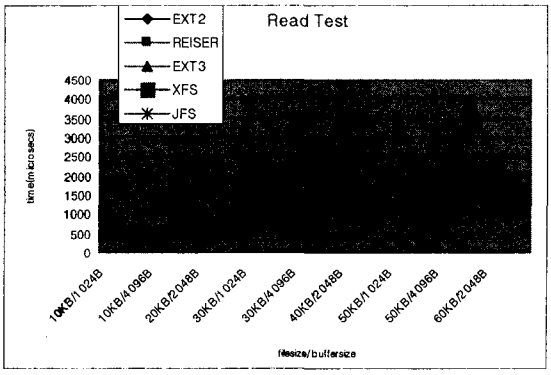
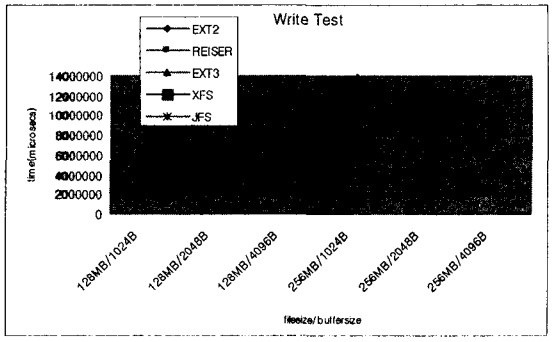
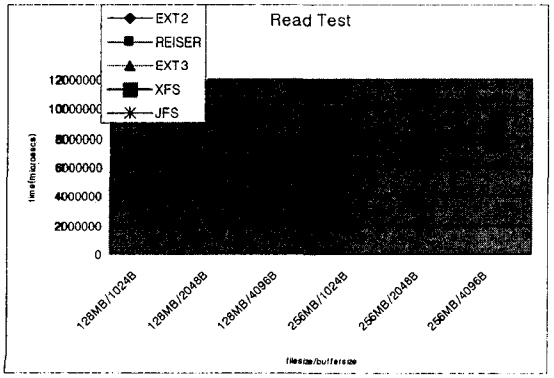


그림 1. 10~60KB 파일에 대한 실험결과





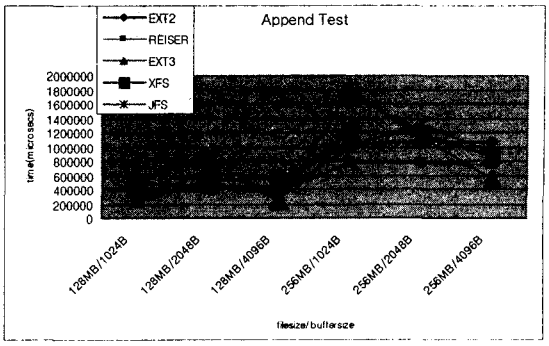


그림2. 128, 256MB 파일에 대한 실험결과

위의 결과는 쓰레드 개수를 하나로 설정했을 때의 결과이다. 쓰레드가 많아질수록 그래프의 모양은 규칙성을 찾기 힘들기 때문에 본 문에서는 생략한다. 쓰기보다 읽기에서 그래프의 기울기가 더 급하게 나타나는데 이것은 프로세서 자체 캐시의 영향이 크기 때문이다. 즉, 10KB~60KB사이의 파일은 프로세서의 캐시보다 작은 크기의 파일들로 캐시의 영향을 크게 받는다. 128MB와 256MB 파일에 대한 실험에서는 레코드의 크기가 4096B에서 동일한 크기의 파일에 대해 가장 좋은 성능을 보여주고 있는데, 이는 본 실험에서 사용한 시스템에 한정된 결과이며 시스템의 메모리와 레코드 크기에 따라 그 최적 성능이 달라진다.

그래프에서 보는 바와 같이 실험 항목마다 최적의 성능을 보여주는 파일시스템은 모두 다르지만, XFS는 모든 실험 항목에서 고른 성능을 보여준다.

5. 결 론

본 실험은 각 파일시스템에 대한 절대적인 읽기/쓰기 성능 시험을 위한 것은 아니며, 파일시스템의 성능을 상대적으로 비교, 평가하는데, 그 목적이 있다. 이러한 관점에서 XFS는 신뢰성 있는 성능을 보여주고 있다. 향후에 저널링 성능 측정을 위한 복구 실험은 파일시스템에 따라 정확한 파일의 재 시작 속도와 복구 정도를 측정하기 위한 보다 세분화된 정량적 측정 방법에 관한 연구가 필요하다.

참고문헌

[1] "Sentry24", http://www.emstone.com
[2]Stephen C. Tweedie, "Journaling the Linux ext2fs Filesystem", LinuxExpo'98, 1998
[3]*Juan I. Santos, "Journaling File Systems"*, http://linuxgazette.fst.or.kr/issue55/florido.html, July, 2000
[4]Stephen Tweedie, "EXT3, Journaling Filesystem", Linux Symposium, 20 July, 2000
[5]Hans T. Reiser, "ReiserFS", http://www.namesys.com/res_whol.shtml, Jan. 2001
[6]"XFS: A high-performance journaling file system", http://oss.sgi.com/xfs
[7]"JFS for Linux", http://www-124.ibm.com/developerworks/oss/jfs