파일 시스템의 관리

**1. 파일 시스템의 개요와 종류**

* 개요
  + 운영체제가 파일을 시스템의 디스크상에 구성하는 방식이다.
  + 컴퓨터에서 파일이나 자료를 쉽게 발견 및 접근할 수 있도록 보관 또는 조직하는 체계이다.
  + 하드 디스크나 CD-ROM와 같은 물리적 저장소를 관리한다.
  + 파일 서버상의 자료로의 접근을 제공하는 방식과 가상의 형태로서 접근 수단만이 존재하는 방식(procfs 등)도 파일 시스템의 범위에 포함한다.
  + 리눅스 파일 시스템의 구조는 다음과 같이 나타낼 수 있다.
    - | Boot block | block group 0 | --- | block group n-1 | block group n |
    - Block group => | super block(1 block) | group describtors(n blocks) | block bitmap(1 block) | inode bitmap(1 block) | inode table(n blocks) | data block(n blocks) |

|  |  |
| --- | --- |
| **Super block** | 해당 파일 시스템 관련 정보 저장  -블록의 크기(1KB, 2KB, 4KB)  -총 블록의 개수와 블록 그룹의 개수  -inode의 개수 |
| **Group**  **describtor table** | 각 block group을 관리하는 정보 저장  -block bitmap와 inode bitmap의 블록 번호  -첫번째 inode table block의 블록 번호  - 그룹 안에 있는 빈 블록, 그룹 안에 있는 inode 수, 그룹 안에 있는 빈 디렉터리 수 |
| **Block bitmap** | 그룹 내에 있는 각 블록의 사용 상태를 나타냄 |
| **inode** | 파일에대한 제어 정보 및 데이터 블록 포인터 저장  파일의 이름을 제외한 해당 파일의 모든 정보를 저장  파일 이름에 부여되는 고유번호, 파일 형태, 크기, 위치, 파일의 소유자 등  모든 파일과 디렉터리들은 각각 1개의 inode를 할당  Inode bitmap: 이 블록에 속한 각 비트는 그룹 내에 있는 각 inode의 사용상태를 나타냄  Inode table: 각각의 inode에 대한 정보를 나타내는 inode describtor로 구성 |

* 종류
  + 리눅스 전용 디스크 기반 파일 시스템

|  |  |
| --- | --- |
| **파일 시스템** | **설명** |
| **ext**  **(ext1)** | 리눅스 초기에 사용되던 파일 시스템이며 호환성이 없음  ext2의 원형  2GByte의 데이터와 파일명을 255자까지 지정 가능  파일 접근에 대한 타임 스탬프, 아이노드 수정 지원불가 |
| **ext2** | ext파일 시스템의 다음 버전  고용량 디스크 사용을 염두하고 설계된 파일 시스템  쉽게 호환되며 업그레이드도 쉽게 설계되어 있음 |
| **ext3** | ext2의 확장판  리눅스의 대표적인 저널링(jornaling)을 지원하도록 확장된 파일 시스템  ACL(Access Control List)을 통한 접근 제어 지원 |
| **ext4** | ext2 및 ext3와 호환성이 있는 확장 버전  파일에 디스크 할당 시 물리적으로 연속적인 블록을 할당  -파일 접근 속도 향상 및 단편화를 줄이도록 설계된 파일 시스템  64비트 기억공간 제한을 없앰  16TeraByte의 파일을 지원  -파일 확장자는 필요 없으며 파일 특성을 알리기 위해 확장자 사용 가능  -‘.’으로 시작하는 파일은 숨겨진 파일  -‘.’은 현재 디렉터리, ‘..‘은 부모 디렉터리 |

* 비교

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Ext2** | **Ext3** | **Ext4** |
| **개발년도/커널버전** | 1993년 | 2001년 커널 2.4.15 | 2006년커널 2.6.19  2008년커널 2.6.28 |
| **최대 파일 크기** | 16GB~2TB | 16GB~2TB | 16GB~2TB |
| **최대 파일 시스템 크기** | 2TB~32TB | 2TB~32TB | 1EB |
| **특징** | 저널링 기능 없음 | 저널링 기능 포함 | 확장 멀티블록 할당 |

* 저널링 파일 시스템
  + 시스템의 비정상적인 종료 시 저널(로그)을 이용하여 빠르면서도 안정적인 복구가 가능하다.
  + 데이터를 디스크에 쓰기 전에 로그에 데이터를 남겨 시스템의 비정상적인 셧다운에도 로그를 사용해 빠르고 안정적인 복구 기능을 제공하는 기술이다.
* 기존의 ext2파일의 경우에는 시스템이 동작을 멈추기 직전에 파일 시스템에 수정을 가하고 있었는지 전혀 알 수 없다.
* 저널 기능이 없는 경우 시스템을 복구하기 위해서 fsck에 의해 관리되는 슈퍼블록, 비트맵, 아이노드 등을 모두 검사해야 하기 때문에 시간이 오래 걸린다.
* 운영형태
* 저널이라는 로그에 시스템 전 상태를 저장한다.
* 시스템의 비 정상적인 종료 시 저널(로그)를 검사한다.
* 저널(로그)를 정보를 바탕으로 파일 시스템에 수정된 내용을 적용한다.
* 저널링 기술이 적용된 파일 시스템: JFS, xfs, ResierFS
* 네트워크 파일 시스템: SMB, CIFS(SMB를 확장 파일 시스템), NFS,
* 기타 지원이 가능한 파일 시스템: FAT, VFAT, FAT32, NTFS, ISO 9660, UDF, HPFS

**2. 관련 명령어**

* 명령어 mount와 umount
  + 마운트는 특정 디바이스를 특정 디렉터리처럼 사용하기 위해 장치와 디렉터리를 연결한다.
  + 리눅스는 PnP(Plug and Play) 기능을 지원하지만 지원하는 하드웨어가 많지 않으므로 시스템 부팅 후에 수동으로 마운트해서 사용을 하고 사용이 끝난 후에는 언마운트를 시킨다.
  + 마운트 설정 명령어 형식은 다음과 같다.
    - 형식: mount [옵션] [디바이스명] [디렉터리명]

|  |  |  |
| --- | --- | --- |
| **옵션** | | **설명** |
| **-a** | | /etc/fstab 파일에 정의된 모든 시스템 마운트 |
| **-n** | | /etc/mtab 파일에 정보를 저장하지 않고 마운트 |
| **-f** | | 실제 마운트하는 것이 아니라 마운트가 가능한지를 테스트 |
| **-t** | **파일 시스템** | vfat: 마이크로 소프트 파일 시스템 FAT-32를 마운트  ext2, ext3, ext4: 리눅스 파일 시스템 마운트  ios9660: CD-ROM이나 DVD마운트  smbfs: 삼파 파일 시스템 마운트  nfs: 네트워크 파일 시스템인 공유된 영역을 마운트 |
| **-o** | **추가 설정** | ro: 읽기 전용으로 마운트  rw: 읽기/쓰기 모드로 마운트  loop: Loop 디바이스나 CD-ROM 이미지 파일 iso 마운트  remout: 파티션을 재마운트  noatime: 파일이 변경되기 전까지 access time이 변경하지 않음  sync: 파일 시스템에 대한 입출력을 동기화  user: 일반 사용자가 마운트 |

* 파일 /etc/mtab은 현재 마운트된 블록 시스템 정보를 표시한다.
  + 시스템 부팅 시에는 파일 /etc/fstab의 마운트 정보를 참조한다.
  + 파일 /etc/mtab에서는 현재 시스템의 마운트 정보를 확인 할 수 있다.
* 마운트 해제 명령어 형식은 아래와 같다.
  + 형식: umount [옵션] [디바이스명] [디렉터리명]

|  |  |
| --- | --- |
| **옵션** | **설명** |
| **-a** | 파일 /etc/fstab에 명시된 파일 시스템을 마운트 해제 |
| **-n** | 파일 /etc/mtab 파일을 갱신하지 않고 마운트 해제 |
| **-t** | 언마운트할 파일 시스템을 지정 |