[Chapter 13]

소프트웨어융합학과/2020105742/한지훈

1. File system
   A. File 정의: secondary storage에 저장되는, 연관된 data를 묶어놓은 단위
      I. 관리: file system
      II. 특징: 고유 이름 있음, byte 단위로 표현
   B. 기능
      I. abstraction for secondary storage: file은 추상화된 최소 단위
      II. Organize files: 디렉토리를 통해 파일들을 정리 가능해야 함
      III. Sharing: data를 클라우드 등을 통해 다른 사람들과 공유할 수 있어야 함
      IV. Protection: file들에 대한 보호 필요함

2. File Attributes
   A. 개요: File system이 각각의 파일을 관리하기 위해서 가지고 있는 각종 정보들
   B. 종류: 접근 가능한 사용자/방식, 비밀번호, 생성자 등

3. File Types
   A. 종류
      I. file system이 이해: file system 자체가 알고 있으며, 사용하는 파일들
         - 종류: directory, symbolic link(바로가기) 등
      II. OS or runtime libraries이 이해
         - 2개 간 구분 필요: 커널 자체는 이해 X, 시스템 라이브러리의 컴파일러 설치해야 코드 인식하는 경우
            ➢ 유저와 커널/라이브러리 간 약속으로 이해하는 파일
         - 종류: 실행파일(.exe), 소스코드 등
      III. Application이 이해
         - 특정 어플리케이션을 설치 시 해당 파일을 사용할 수 있는 방식
         - 종류: jpg, avi, mp3 등
   B. 경계선(다 커널/file system 이해하게 하기): 개발자 마음, 사용 목적이 따라 사용
   C. File type encoding
      I. 윈도우: 파일의 목적을 서로 구분하기 위한 확장자명이 필수적
      II. 리눅스: 확장자 자체는 중요 X, 여러 사용자들에게 어떤 file들을 허용할지 말지가 더 중요

4. File Access 방식
   A. Sequential access: 앞에서부터 순차적으로 접근하는 가장 기본적이며, 단순한 방식
   B. Direct access(Random access): 모든 요소를 동일한 시간에 접근
      I. 조건
         - Record access: 메모리 크기 < 디스크 크기이므로 bit단위보다는 레코드 단위 access가 필요
         - Index access: 결국 모든 레코드에 대한 Random access 위해서는 인덱스 필요
            ● 키를 포함한 Index file 따로 생성 시 성능 더 높아짐

5. Directories
   A. 개요: 여러 개의 file의 리스트를 가질 수 있는 특수한 file
      ➢ 경로(hierarchical) 생성, 절대경로/상대경로로 구분

6. Pathname Translation
   A. 정의: 경로를 텍스트로 작성하여 code로 표현하는 방법
      I. 윈도우: 구분자가 back slash라서 경로 표시 할 때 두번 써야 함
      II. 리눅스: 구분자가 /(slash)

7. File System Mounting
   A. 개요: file system을 커널에 물리는 방식
      I. 윈도우: drive letter 사용(C:\, D:\ 등)
      II. 리눅스: 디스크의 root를 커널의 root 아래에 속하도록 붙힘(mount)
         - 커널로부터 디스크(file system)에 접근 가능
         - 여러 사람 사용시 사용 시나리오가 서로 다르기 때문에 mount/unmount 통해 관리함

8. Remote File Systems
   A. 개요: 네트워크를 통해 내 컴퓨터의 File system인 것처럼 사용
      I. 윈도우: CIFS(Common Internet File System) 사용
      II. 리눅스: NFS(Network File System) 사용
      III. DNS 등의 분산시스템을 사용하는 경우도 있음

9. Protection
   A. 개요: 파일의 여러 요소에 대한 보호
   B. 윈도우: 관련 기능들 있긴 하지만 보통 혼자 사용하기 때문에 크게 필요하지 않음
   C. 리눅스: 보통 여러 사용자가 공유하여 사용하기 때문에 protection 필요함
      I. ACL: object(file)단위로 관리 – 한 개의 file에 대한 모든 사용자들의 권한 관리
         - 단점: file 수 너무 많아지면 문제 발생 가능
      II. Capability: subject(유저)단위로 관리 – 한명의 유저에 대한 모든 파일들의 권한 관리
         - 장점: file 공유 시에는 한번에 할 수 있기 때문에 성능 높음
      III. 비교:
         - 권한도 data이며, 유저 수보다 파일 수가 훨씬 많음
            - 또한 유저는 보통 file 중 일부만 접근하기 때문에 접근 안할 파일에 대한 정보 저장 필요 X
            - ACL이 읽을 file에 대한 권한만 확인할 수 있기 때문에 메모리 낭비가 적어서 좋음
            - 결국 디스크 접근 시 file 단위로 접근하기 때문에 발생하는 차이

10. Access Lists
   A. Access 종류: 3가지(read, write, execute) -> 3bit(3자리 8진수)로 표현
   B. 유저 종류: 3가지(owner(root 권한), group(특정 그룹에 대한 권한), public(모든 사람의 권한))
   C. 예시: chmod 761 game
      I. Owner: 7 -> 111 -> read, write, execute 모두 할 수 있음
      II. Group: 6 -> 110 -> read, write만 할 수 있음
      III. Public: 1 -> 001 -> execute만 할 수 있음

11. Memory-Mapped File
    A. 개요: logical memory 일부를 disk의 file에 매핑
        I. 기존 file: 안정적이긴 하지만 사용시 open() 등 해야하기 때문에 복잡함
        II. 기존 file 대비 장점: read(), write() 등의 system call 없이 file을 사용할 수 있음