

이화여자대학교

EWHA WOMANS UNIVERSITY

전체화면으로 종료하려면 [F11] 키를 누르세요.

File and File System

File

"A named collection of related information"

일반적으로 비휘발성의 보조기억장치에 저장

운영체제는 다양한 저장 장치를 file이라는 동일한 논리적 단위로 볼 수 있게 해 줌

Operation (연산)

create, read, write, reposition (seek), delete, open, close 등

File attribute (혹은 파일의 metadata)

파일 자체의 내용이 아니라 파일을 관리하기 위한 각종 정보들

파일 이름, 유형, 저장된 위치, 파일 사이즈

접근 권한 (읽기/쓰기/실행), 시간 (생성/변경/사용), 소유자 등

File system

운영체제에서 파일을 관리하는 부분

파일 및 파일의 메타데이터, 디렉토리 정보 등을 관리

파일의 저장 방법 결정

파일 보호 등

root 디렉토리부터 계층구조를 함.

→ 구조에 따른.

File: 이름을 통해서 접근하는 단위.

저장 장치 관리를 위해 File이란 단위로 관리  
(ex.) Disk1, Disk2...-)  
devices special file

## Operator

reposition.: 다른 부분까지 참조할 땐 쓰는 연산.  
(보통 순차탐색)

open, close: 읽고 쓰기 위해 open, close 필요.

이 open, close 연산 왜 필요?!

open: file의 메타데이터를 얻는 것  
(파일 관리에 필요한 데이터)

이화여자대학교

EWHA WOMANS UNIVERSITY

전체화면으로 종료하려면 [F11] 키를 누르세요.

Directory and Logical Disk

Directory

파일의 메타데이터 중 일부를 보관하고 있는 일종의 특별한 파일

그 디렉토리에 속한 파일 이름 및 파일 attribute들

operation

search for a file, create a file, delete a file

list a directory, rename a file, traverse the file system

Partition (=Logical Disk)

하나의 (물리적) 디스크 안에 여러 파티션을 두는게 일반적

여러 개의 물리적인 디스크를 하나의 파티션으로 구성하기도 함

(물리적) 디스크를 파티션으로 구성한 뒤 각각의 파티션에 file system을 깔거나 swapping 등 다른 용도로 사용할 수 있음

Directory file의 내용.

디렉토리 밑에 있는 파일이 어떤 건지. 메타데이터를 내용으로 하는 파일.

운영체제가 보는 디스크 = 논리적 디스크 = partition.

- file system 용도.

- swapping 용도.

이화여자대학교

EWHA WOMANS UNIVERSITY

전체화면으로 종료하려면 [F11] 키를 누르세요.

open()

retrieves metadata from disk to main memory

File metadata

File metadata

File Content

Pointers 저장.

## open ( )

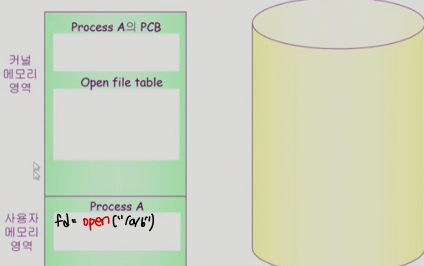
- open("/a/b/c")
  - ✓ 디스크로부터 파일 c의 메타데이터를 메모리로 가지고 올
  - ✓ 이를 위하여 directory path를 search
    - 루트 디렉토리 "/"를 open하고 그 안에서 파일 "a"의 위치 획득
    - 파일 "a"를 open한 후 read하여 그 안에서 파일 "b"의 위치 획득
    - 파일 "b"를 open한 후 read하여 그 안에서 파일 "c"의 위치 획득
    - 파일 "c"를 open한다
  - ✓ Directory path의 search에 너무 많은 시간 소요
    - Open을 read / write와 별도로 두는 이유임
    - 한번 open한 파일은 read / write 시 directory search 불필요
  - ✓ Open file table
    - 현재 open 된 파일들의 메타데이터 보관소 (in memory)
    - 디스크의 메타데이터보다 몇 가지 정보가 추가
      - Open한 프로세스의 수
      - File offset: 파일 어느 위치 접근 중인지 표시 (별도 테이블 필요)
  - ✓ File descriptor (file handle, file control block)
    - Open file table에 대한 위치 정보 (프로세스 별)

그러는 file open.  $\Rightarrow$  (이 메타데이터가 메모리로 올라온다.)

그러는 directory 경로가 지정해준 구조에 있는 "a" open하면 거기 저장되어 있는 지 어떻게 찾을까?

root directory 위치는 알려져 있기 때문에 따라 내려가서 (위치 찾을).

## open ( )



• open은 시스템 콜 (I/O)

open 과정.

1. open

2. CPU가 운영체제에 요청.

3. root의 메타데이터 open.

4. root의 실제 위치 찾을. (메타데이터 저장되어 있음)

5. root의 content. (root는 디렉토리 파일이기 때문에 디렉토리 및 파일들의 메타데이터 있음)

6. root 안에 a 파일의 메타데이터 찾을. (open)

7. a의 메타데이터를 file system 위치 정보.

a로 디렉토리 확인함.  $\rightarrow$  b라는 메타데이터가 있다.

8. b open.

open system call의 리턴값.

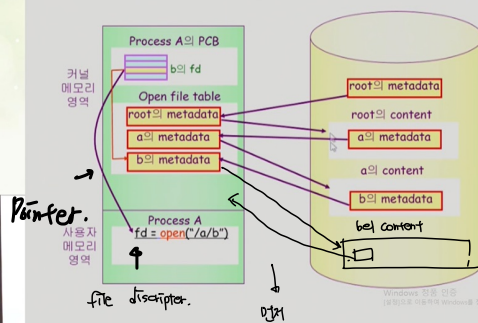
: 각 프로세스마다 open한 file들에 대한 메타데이터 포인터를 가지고 있는 일종의 배열이 정의되어 있음.

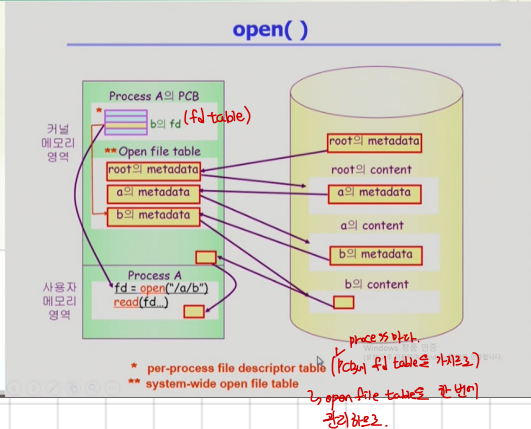
지금 open은 pointer가 배열의 인덱스에 만들어지고

1 배열에서 몇 번째 index냐.  $\rightarrow$  file의

descriptor가 되고 그 값을 사용자 프로그램에게 return하게 됨

## open ( )





구현에 따라 table이 3종류일수도 있음.

why? metadata가 disk에 있을 경우... 이런 내용임.

memory에 있으면 휘발성 메모리이므로

필요. 어느 위치 있고 있는지 알 수

있는 offset table 필요.

그래서 open file table 이 2개 두는 것이 일반적

9. **read**(fd...) file descriptor를 통해 read, write 가능.

10. read는 system call이 다시 운영체제로 cpu가 넘어감.

11. 해당 fd에 대응하는 파일의 메타데이터 부분을 open file table에서 따라 간 다음, 위치 정보를 얻고 시작위치부터 읽는다. (이후 요청사항이 read.)

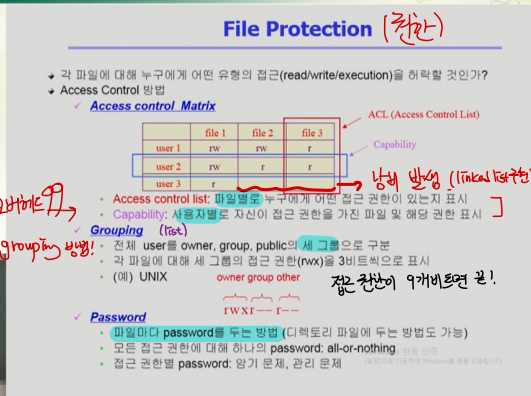
12. b의 내용을 사용과 프로그램에게 직접주는게 아닌, 운영체제가 자신의 버퍼에 읽어들여 놓는다.

13. 그 내용을 사용과 프로그램에게 copy하여 전달. (read 작업 끝)

→ 만약 다른 프로그램이 동일 파일의 내용을 요청하면 다시 읽는 필요없이 OS의 메모리에 있는 내용 전달. file system의 buffer cache.

→ 시스탬콜러지에 요청 내용이 있는 것은 OS로 넘어감.

즉, OS가 모든 버퍼를 읽고 read, write 요청을 사용자에게



그럼에도 왜냐하면?

여러 grouping 방법!

memory protection = 메모리는 프로그램마다 별도로 들으니까 자식만 참조가능

file은 여러 사용자, 여러 프로그램이 사용가능.

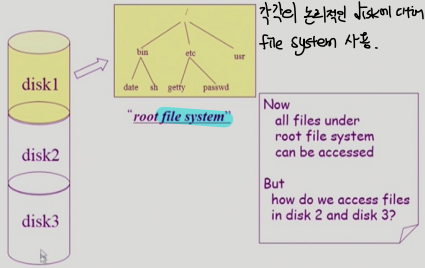
따라서 1. 접근 권한이 누구에게 있는가.

2. 접근 권한이 어떤것이 가능한가?

(r, w, e)

고려.

## File System의 Mounting

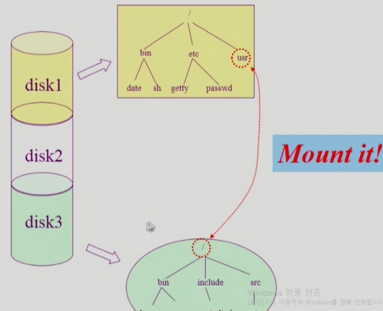


Windows: 윈도우 인스턴스  
(파일)으로 이동하거나 Windows를 종료하십시오.

만약 다른 partition에 설치된 file system에 접근해야 한다면?

Mounting 연산 제공.

## File System의 Mounting



Windows: 윈도우 인스턴스  
(파일)으로 이동하거나 Windows를 종료하십시오.

Mount된 directory에 접근하게 되면 또 다른 file system의 root directory도 접근하는 것임.

## Access Methods

- 시스템이 제공하는 파일 정보의 접근 방식
  - 순차 접근 (sequential access)
    - 아세트 테이프를 사용하는 방식처럼 접근
    - 읽거나 쓰면 offset은 자동적으로 증가
  - 직접 접근 (direct access, random access) (임의 접근)
    - LP 레코드 판과 같이 접근하도록 함
    - 파일을 구성하는 레코드를 임의의 순서로 접근할 수 있음

Windows: 윈도우 인스턴스  
(파일)으로 이동하거나 Windows를 종료하십시오.