5강. 기본 운영(2)

- 파일 및 디렉터리의 권한 변경 명령(change mode)
- 사용법 : chmod [옵션] 권한 파일명...
  - -R : 하위 경로에 있는 파일들과 디렉터리까지 권한을 변경
  - 권한

-읽기 (r) : 파일 및 디렉터리 내용 보기
-쓰기 (w) : 파일의 쓰기 및 디렉터리 안에 파일 생성 및 삭제
-실행 (x) : 파일 실행 및 디렉터리 접근

- 파일 및 디렉터리의 권한 표시

-사용자 권한(u) : 첫 번째 열 (예: rwx)
-그룹 권한(g) : 두 번째 열 (예: r-x)
-다른 사용자들 권한(o) : 세 번째 열 (예: r-x)

- chmod 명령에서 권한 지정 방법(8진수 모드)
- ugo 각각에 rwx 권한을 지정함
- 예 : chmod 755 test
- 허가는 1, 불허는 0으로 하여 2진수로 표시한 후 8진수 모드로 권한 지정
- u에 rwx, g와 o에 rx를 권한을 지정한다면, rwxr-xr-x -> 111101101 -> 755

- chmod 명령에서 권한 지정 방법(기호 모드)
- 형식 : [ugoa][+-=][rwx]
  - ugoa : 각각 user, group, other, all을 의미, 생략하면 all
  - +-= : 각각 권한 추가, 권한 삭제, 권한 지정을 의미
  - rwx : 각각 read, write, execute를 의미, = 뒤에서 생략되면 아무 권한도 없음을 의미
  - 예 : chmod go-rx test (그룹과 다른 사용자의 기존 권한에서 r과 x를 제거)
  - 예 : chmod u=rwx,g=rx,o= java (사용자에게 rwx, 그룹에게 rx 권한을 부여)

- 리눅스에서 파일 속성 변경(change attributes)
- 사용법 : chattr [옵션] [+-=속성] 파일명...
  - -R : 하위 경로에 있는 파일들과 디렉터리까지 속성 변경
  - +-= : 각각 속성 추가, 속성 삭제, 속성 지정을 의미
  - i : 읽기 전용 모드, root 만 속성 변경 가능
  - a : 추가 전용 모드. root 만 속성 변경 가능
  - 예 : chattr +i test (읽기 전용 파일로 변경)
- 용도

-파일의 소유자도 읽기 전용(또는 추가 전용)으로만 파일을 열 수 있음
-root도 설정을 해제하지 않으면 파일을 지울 수 없게 지정
-lsattr 명령으로 속성 확인 가능

- 파일 복사 명령(copy)

- 사용법 : cp [옵션] 복사할파일명 대상파일명
- 사용법 : cp [옵션] 복사할파일명... 대상디렉터리명
- 사용법 : cp -r [옵션] 복사할디렉터리명 대상디렉터리명
- 옵션
  - -a : 파일의 속성, 링크 정보들을 유지하면서 복사
  - -b : 파일이 존재하면 백업을 만들고 복사
  - -f : 기존의 파일을 강제로 삭제하고 복사
  - -i : 파일이 존재하면 덮어쓰기 여부를 확인
  - -r 또는 -R : 디렉터리를 복사
- 사용 예
  - cp test1.txt test2.txt (현재 디렉터리의 test1.txt 파일을 test2.txt로 복사)
  - cp /home/nipark/test.txt (/home/nipark 디렉터리의 test.txt 파일을 현재 디렉터리로 복사)
  - cp -r * backup (현재 디렉터리의 숨김 파일을 제외한 모든 파일과 디렉터리를 backup 디렉터리로 복사)

- 파일 삭제 명령(remove)
- 사용법 : rm [옵션] 파일명...
  - -i : 파일 삭제 여부를 확인
  - -f : 파일을 강제로 삭제
  - -r : 디렉터리인 경우 하위 경로에 있는 파일들과 디렉터리까지 삭제
- 사용 예
  - rm -r 디렉터리명 (디렉터리를 하위 디렉터리 및 파일과 함께 삭제하는 방법)
  - rm -f *.tmp (현재 디렉터리의 .tmp로 끝나는 파일을 모두 강제로 삭제)

- 파일의 이름을 변경하거나 다른 디렉토리로 파일을 이동시키는 명령(move)
- 사용법 : mv [옵션] 복사할파일명 대상파일명 (현재의 디렉토리 안에서 파일 이름 변경)
- 사용법 : mv [옵션] 복사할파일명... 대상디렉터리명 (파일(들)을 지정한 디렉터리로 이동)
- 옵션
  - -i : 같은 이름이 존재할 경우 덮어쓰기 여부를 확인
  - -f : 강제로 이동
- 사용 예
  - mv test1.txt test2.txt (현재 디렉터리의 test1.txt 파일을 test2.txt로 이름 변경)
  - mv /home/nipark/test.txt /home/nipark/Sub (/home/nipark 디렉터리의 test.txt 파일을 /home/nipark/Sub 디렉터리로 이동)

- 파일을 다른 이름으로 연결하는 명령 (link)
- ln 원본파일명 대상파일명

-하드 링크(hard link)를 만드는 것
-하나의 파일에 여러 개의 이름을 부여
-다른 파일 시스템 간에는 링크할 수 없음

- ln -s 원본파일명 대상파일명

-심볼릭 링크(symbolic link)를 만드는 것
-다른 파일을 가리키는 역할로 윈도우의 '바로 가기'와 같음
-다른 파일 시스템에 링크를 만들 수 있음

- 사용 예
  - ln test mytest; ls -l (현재 디렉터리의 test 파일에 myhard라는 하드 링크를 생성 후 확인)
  - ln -s test mysymbolic; ls -l (현재 디렉터리의 test 파일에 mysymbolic이라는 심볼릭 링크를 생성 후 확인)

● 조건에 맞는 파일 및 디렉터리를 찾는 명령(find)

- 사용법 : find [옵션1] [경로] [옵션2] [표현식]
- 옵션
  - 경로 : 검색할 위치, 하위 디렉터리도 검색
  - -L : 옵션1, 심볼릭 링크된 디렉터리도 검색
  - -maxdepth n : 옵션2, 검색할 디렉터리 깊이, 1이면 하위 디렉터리를 검색하지 않음
  - -mount : 옵션2, 경로가 위치한 곳과 다른 파일 시스템은 검색하지 않음
- 표현식은 다음 요소로 구성됨
  - test : 파일을 찾는 조건
  - action : 조건에 일치하는 파일에 대해 수행할 작업
  - operator : 조건을 연산자로 조합
- 표현식의 요소 : test
  - -name : 파일 이름으로 찾음.
  - -perm : 파일의 접근 권한으로 찾음
  - -user : 파일의 소유자로 찾음
  - -group : 파일의 그룹으로 찾음
  - -type : 파일의 유형, 파일(f), 디렉터리(d), 링크(l) 등
  - -mtime +n|-n : 변경된 날짜
  - -mmin +n|-n : 변경된 시간(분)
- 표현식의 요소 : action
  - -print : 파일 이름을 화면에 출력
  - -fprint file : 파일 이름을 파일에 출력
  - -exec : 파일에 특정 명령을 수행, 주로 -exec command { } \; 형식
- 표현식의 요소 : operator
  - ! 표현식 : NOT인 경우
  - 표현식1 표현식2 : AND인 경우
  - 표현식1 -o 표현식2 : OR인 경우
- 사용 예
  - find / -name passwd -print (/ 이하에서 이름이 passwd인 파일을 찾아 화면 출력)
  - find /etc -name passwd -exec cat { } \; (/etc 이하에서 이름이 passwd인 파일을 찾아 각각의 내용을 화면 출력)

◦ find /sbin -type l -print (/sbin 이하에서 심볼릭 링크를 찾아 화면 출력)
◦ find /var -user nipark -mtime -2 -print (/var 이하에서 소유자가 nipark이고 수정 시간이 2일이 안된 파일을 찾아 화면 출력)
◦ find /home -name '*.bak' -exec rm -f { } \; (/home 이하에서 이름이 .bak로 끝나는 파일을 찾아 강제로 삭제함)
◦ find ~ -size 0 -ok rm { } \; (홈디렉터리 이하에서 파일 사이즈가 0인 것을 찾아 삭제)

● 다음 문제에 대한 정답을 서술하시오.

17. 링크를 부여하는 방법으로는 하드 링크와 심볼릭 링크가 있다. 각각의 링크 방법에 대하여 설명하여 보자.

① 하드 링크 (Hard Link)

하드 링크는 하나의 파일에 여러 개의 이름을 부여하는 것이다. 다시 말해, 같은 파일을 이름만 다르게 하여 부른다는 뜻이다. 그 파일을 없애려면 링크로 생성된 링크 파일을 모두 지워야 한다. 또한, 같은 i-node 번호를 가지기 때문에 다른 파일 시스템 간에는 링크할 수 없다.

② 심볼릭 링크 (Symbolic Links)

심볼릭 링크는 윈도즈 운영체제의 '바로 가기'와 유사하다. 링크로 생성된 파일의 내용은 존재하지 않으며, 각각의 i-node를 가진 또 다른 파일이 어디를 가리키고 있는지 알려주는 역할을 한다. 심볼릭 링크를 만들 경우 하드 링크와는 다르게 링크를 다른 곳으로 이동시키면 링크가 깨져서 사용할 수 없다. 또한, 다른 파일 시스템 간에 생성할 수 있어 다른 파티션에 링크 파일을 만들 수 있다.