4주차 2차시 사용자 관리

【학습목표】

1. 리눅스 사용자 정보에 대해 설명할 수 있다.
2. 리눅스 사용자 정보 관련 함수를 사용할 수 있다.

학습내용1 : 사용자 정보 개요

1. 사용자 관리

① 개요
리눅스는 다중 사용자 시스템이므로 사용자를 구별하고 사용자에게 적절한 자원을 할당해주는 방법이 필요
사용자 계정은 사용자가 시스템에 접근할 수 있는 유일한 방법
시스템 관리자의 입장에서도 사용자의 접근 권한을 통제할 수 있는 중요한 수단

② 사용자 계정 관련 파일
* /etc/passwd 파일
사용자 계정 정보가 저장된 기본 파일
한 행에 사용자 한 명에 대한 정보가 기록되며, 쌍점(:)으로 구분되는 일곱 개의 항목으로 구성

로그인ID : x : UID : GID : 설명 : 홈 디렉터리 : 로그인 쉘
ⓐ ⓑ ⓒ ⓓ ⓔ ⓕ ⓖ

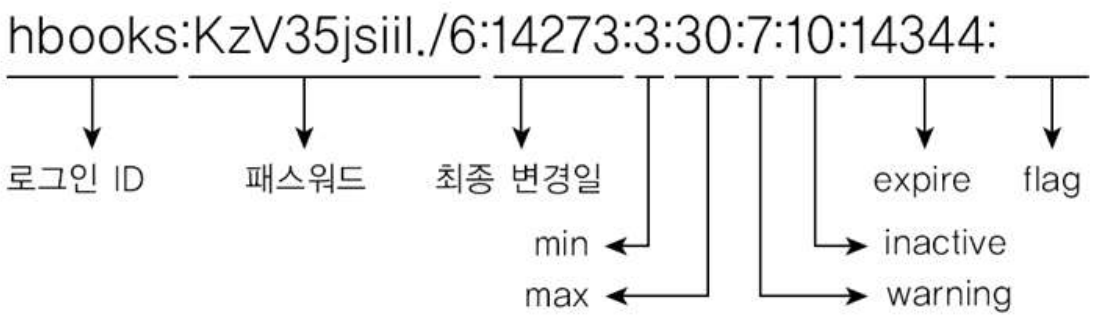
* 로그인 ID: 사용자 계정의 이름, 32자를 넘을 수 없으나 8자로 제한하는 것이 좋다
* x : 초기 유닉스 시스템에서 사용자 암호를 저장하던 항목, 요즘은 /etc/shadow 파일에 별도로 보관
* UID : 사용자 ID 번호로 시스템이 사용자를 구별하기 위해 사용하는 번호
* 0~999번과 65534번은 시스템 사용자를 위한 UID로 예약(0: root, 1: daemon, 2: bin, 7: lp 등)
일반 사용자들은 UID 1000번부터 할당
* 로그인 ID가 다르더라도 UID가 같으면 리눅스 시스템은 같은 사용자로 판단, 따라서 UID가 중복되지 않았는지 주의해야 함
* GID : 그룹 ID, 시스템에 등록된 그룹에 대한 정보는 /etc/group 파일에 저장
* 설명 : 사용자의 실명이나 부서명, 연락처 등 사용자에 대한 일반적인 정보가 기록
* 홈 디렉터리 : 사용자 계정에 할당된 홈 디렉터리의 절대 경로를 기록
* 로그인 셸 : 사용자의 로그인 셸을 지정, 우분투에서는 배시 셸(/bin/bash)을 기본 셸로 사용

- passwd 구조체

```
struct passwd {
        char   *pw_name;
        char   *pw_passwd;
        uid_t   pw_uid;
        gid_t   pw_gid;
        char   *pw_age;
        char   *pw_comment;
        char   *pw_gecos;
        char   *pw_dir;
        char   *pw_shell;
};
```

* /etc/shadow 파일
사용자 암호에 관한 정보를 별도로 관리하는 파일
root 계정으로만 내용을 볼 수 있음

```
# cat /etc/shadow
root:lyTy6ZkWh4RYw:13892::::::
daemon:NP:6445::::::
bin:NP:6445::::::
......
hbooks:KzV35jsiil./6:14273:3:30:7:10:14344:
```



* /etc/login.defs 파일
사용자 계정의 설정과 관련된 기본 값을 정의

* /etc/group 파일
그룹에 대한 정보가 저장
/etc/passwd 파일의 GID 항목에 지정된 그룹이 기본 그룹이며, 사용자가 속한 2차 그룹은 /etc/group 파일에 지정

* /etc/gshadow 파일
그룹 암호가 저장
리눅스에서만 사용하는 파일(유닉스에는 없음)

③ 사용자 계정 관리 명령 : useradd
사용자 계정 생성.
형식 : useradd [옵션] [login ID]

* 옵션
-u uid : UID 지정
-o : UID 중복 허용
-g gid : 기본 그룹의 GID지정
-G gid : 2차 구룹의 GID 지정
-d 디렉터리명 : 홈디렉터리 지정
-s 쉘 : 기본 쉘 지정

학습내용2 : 사용자 정보 검색 함수

; 사용자 정보, 그룹정보, 로그인 기록 검색
; /etc/passwd, /etc/shadow, /etc/group, /var/adm/utmpx

1. 로그인명 검색 : getlogin(3), cuserid(3)

getlogin() : 현재 프로세스를 실행한 사용자의 로그인명을 리턴
cuserid() : 현재 프로세스의 소유자 정보로 로그인명을 찾아 리턴

성공 시 로그인명, 실패시 NULL 리턴

```
#include <unistd.h>

char *getlogin(void);
```

```
#include <stdio.h>

char *cuserid(char *s);
```

s : 검색한 로그인명을 저장할 주소

2. UID검색 : getuid(2), geteudi(2)

getuid(2) : 현재 프로세스의 실제 사용자 ID
geteuid(2) : 유효 사용자 ID

```
#include <sys/types.h>
#include <unistd.h>

uid_t getuid(void);
uid_t geteuid(void);
```

3. 패스워드 파일 검색

① UID로 passwd 파일 읽기 : getpwuid(3)

```
#include <pwd.h>

struct passwd *getpwuid(uid_t uid);
```

uid : 검색할 UID

② 이름으로 passwd 파일 읽기 : getpwnam(3)

```
#include <pwd.h>

struct passwd *getpwnam(const char *name);
```

name : 로그인명

③ getpwuid 함수 사용하기

```
01  #include <unistd.h>
02  #include <pwd.h>
03
04  int main(void) {
05      struct passwd *pw;
06
07      pw = getpwuid(getuid());
08      printf("UID : %d\n", (int)pw->pw_uid);
09      printf("Login Name : %s\n", pw->pw_name);
10
11      return 0;
12  }
```

```
# ex4_7.out
UID : 0
Login Name : root
```

④ getpwnam 함수 사용하기

```
01  #include <pwd.h>
02
03  int main(void) {
04      struct passwd *pw;
05
06      pw = getpwnam("hbooks");
07      printf("UID : %d\n", (int)pw->pw_uid);
08      printf("Home Directory : %s\n", pw->pw_dir);
09
10      return 0;
11  }
```

```
# ex4_8.out
UID : 100
Home Directory : /export/home/han
```

⑤ /etc/passwd 파일 순차적으로 읽기
getpwent() : 사용자 정보를 순차적으로 읽음. 파일끝에서 NULL 리턴
setpwent() :파일의 오프셋을 파일의 처음으로 이동
endpwent() : 파일을 닫음
fgetpwent() : 파일포인터 인자로 받음.

```
#include <pwd.h>

struct passwd *getpwent(void);
void setpwent(void);
void endpwent(void);
struct passwd *fgetpwent(FILE *fp);
```

fp : 파일 포인터

4. 섀도우 파일 검색

① /etc/shadow 파일 읽기 : getspnam(3)

```
#include <shadow.h>

struct spwd *getspnam(const char *name);
```

spwd 구조체 (/etc/shadow)

```
struct spwd {
        char  *sp_namp;
        char  *sp_pwdp;
        int    sp_lstchg;
        int    sp_min;
        int    sp_max;
        int    sp_warn;
        int    sp_inact;
        int    sp_expire;
        unsigned int sp_flag;
};
```

② getspnam 함수 사용하기

```
01  #include <shadow.h>
02
03  int main(void) {
04      struct spwd *spw;
05
06      spw = getspnam("hbooks");
07      printf("Login Name : %s\n", spw->sp_namp);
08      printf("Passwd : %s\n", spw->sp_pwdp);
09      printf("Last Change : %d\n", spw->sp_lstchg);
10
11      return 0;
12  }
```

```
# ex4_10.out
Login Name : hbooks
Passwd : KzV35jsiil./6
Last Change : 14273
```

③ /etc/shadow 파일 순차적으로 읽기

```
#include <shadow.h>

struct spwd *getspent(void);
void setspent(void);
void endspent(void);
struct spwd *fgetspent(FILE *fp);
```

5. 그룹 정보 검색

① 그룹 ID 검색하기 : getgid(2), getegid(2)

```
#include <sys/types.h>
#include <unistd.h>

gid_t getgid(void);
gid_t getegid(void);
```

/etc/group 파일의 구조

```
# cat /etc/group
root::0:
other::1:root
bin::2:root,daemon
sys::3:root,bin,adm
adm::4:root,daemon
uucp::5:root
......
```

group 구조체

```
struct group {
        char   *gr_name;
        char   *gr_passwd;
        gid_t   gr_gid;
        char   **gr_mem;
};
```

6. 그룹 파일 검색

① /etc/group 파일 검색 : getgrnam(3), getgrgid(3)

```
#include <grp.h>

struct group *getgrnam(const char *name);
struct group *getgrgid(gid_t gid);
```

name : 검색하려는 그룹명
gid : 검색하려는 그룹의 ID

② /etc/group 파일 순차적으로 읽기

```
#include <grp.h>

struct group *getgrent(void);
void setgrent(void);
void endgrent(void);
struct group *fgetgrent(FILE *fp);
```

fp : 파일 포인터

7. 로그인 기록 검색

① 관련 명령어
who 명령 : 현재 시스템에 로그인하고 있는 사용자 정보
last 명령 : 시스템의 부팅 시간 정보와 사용자 로그인 기록 정보

② utmpx 구조체

```
struct utmpx {
    char      ut_user[32];        /* 사용자 로그인명 */
    char      ut_id[4];           /* inittab id */
    char      ut_line[32];        /* 로그인한 장치이름 */
    pid_t     ut_pid;             /* 실행중인 프로세스 PID*/
    short     ut_type;            /* 현재 읽어온 항목의 종류 */
    struct    exit_status ut_exit;/* 프로세스 종료 상태 코드 */
    struct    timeval ut_tv;      /* 해당정보를 변경한 시간 */
    int       ut_session;         /* 세션 번호 */
    int       pad[5];             /* 예약 영역 */
    short     ut_syslen;          /* ut_host의 크기 */

    char      ut_host[257];       /* 원격호스트명 */
};
```

ut_type : 현재 읽어온 항목의 종류
EMPTY(0) : 비어 있는 항목이다.
RUN_LVL(1) : 시스템의 런레벨이 변경되었음을 나타낸다. 바뀐 런레벨은 ut_id에 저장된다.
BOOT_TIME(2) : 시스템 부팅 정보를 나타낸다. 부팅 시간은 ut_time에 저장된다.
OLD_TIME(3) : date 명령으로 시스템 시간이 변경되었음을 나타낸다. 변경되기 전의 시간을 저장한다.
NEW_TIME(4) : date 명령으로 시스템 시간이 변경되었음을 나타낸다. 변경된 시간을 저장한다.
INIT_PROCESS(5) : init에 의해 생성된 프로세스임을 나타낸다. 프로세스명은 ut_name에 저장하고 프로세스 ID는 ut_pid에 저장한다.
LOGIN_PROCESS(6) : 사용자가 로그인하기를 기다리는 getty 프로세스를 나타낸다
USER_PROCESS(7) : 사용자 프로세스를 나타낸다.
DEAD_PROCESS(8) : 종료한 프로세스를 나타낸다.
ACCOUNTING(9) : 로그인 정보를 기록한 것임을 나타낸다.
DOWN_TIME(10) : 시스템을 다운시킨 시간을 나타낸다. ut_type이 가질 수 있는 가장 큰 값이다.

③ 로그인 기록 검색
/var/adm/utmpx 파일 순차적으로 읽기

```
#include <utmpx.h>

struct utmpx *getutxent(void);
void setutxent(void);
void endutxent(void);
int utmpxname(const char *file);
```

file : 교체할 파일명

【학습정리】

1. 사용자 관리
- 리눅스는 다중 사용자 시스템이므로 사용자를 구별하고 사용자에게 적절한 자원을 할당해주는 방법이 필요
- 사용자 계정은 사용자가 시스템에 접근할 수 있는 유일한 방법
- 시스템 관리자의 입장에서도 사용자의 접근 권한을 통제할 수 있는 중요한 수단

2. 사용자 계정 관련 파일
- /etc/passwd 파일
- 사용자 계정 정보가 저장된 기본 파일
- 한 행에 사용자 한 명에 대한 정보가 기록되며, 쌍점(:)으로 구분되는 일곱 개의 항목으로 구성

3.

로그인ID :	x :	UID :	GID :	설명 :	홈 디렉터리 :	로그인 쉘
ⓐ	ⓑ	ⓒ	ⓓ	ⓔ	ⓕ	ⓖ