Section 8.1

- The **Encrypted Password field** in ***/etc/passwd*** is ignored if shadow pass- words have been enabled (which is typical). In this case, the password field in ***/etc/passwd*** conventionally contains the letter x (although any nonempty charac- ter string may appear) like in OpenBSD it contains the letter *, and the encrypted password is instead stored in the shadow password file ***/etc/shadow***
- If this field is empty this means that no password is required to login to this account
- User ID (UID): This is the numeric ID for this user. If this field has the value 0, then this account has superuser privileges. There is normally one such account, with the login name root.
  - *It is possible (but unusual) to have more than one record in the password file with the same user ID, thus permitting multiple login names for the same user ID. This allows multiple users to access the same resources (e.g., files) using different passwords. The different login names can be associated with differ- ent sets of group IDs.*
- **Login Shell** field in ***/etc/passwd*** becomes the value of the ***SHELL*** env variable

Section 8.2: Groups

- In early UNIX implementations, a user could be a member of only one group at a time. A user's initial group membership at login was determined by the group ID field of the password file and could be changed thereafter using the newgrp(1) command, which required the user to supply the group password (if the group was password protected)
- 4.2BSD introduced the concept of multiple simultaneous group memberships, which was later standardized in POSIX.1-1990.

Section 8.4: Retrieving User & Group Info

```c
#include <pwd.h>
struct passwd *getpwnam(const char *name);
struct passwd *getpwuid(uid_t uid);

// Both return a pointer on success, or NULL on error;
// see main text for description of the “not found” case
```

```c
struct passwd {
    char *pw_name;      /* Login name (username) */
    char *pw_passwd;    /* Encrypted password */
    uid_t pw_uid;       /* User ID */
    gid_t pw_gid;       /* Group ID */
    char *pw_gecos;     /* Comment (user information) */
    char *pw_dir;       /* Initial working (home) directory */
    char *pw_shell;     /* Login shell */
};
```

- The ***pw_gecos*** and ***pw_passwd*** fields of the passwd structure are not defined in ***SUSv3***, but are available on all UNIX implementations.
  - *The pw_gecos field derives its name from early UNIX implementations, where this field contained information that was used for communicating with a machine running the General Electric Comprehensive Operating System (GECOS). Although this usage has long since become obsolete, the field name has survived, and the field is used for recording information about the user.*

- All functions that are like **`getpwnam(), getpwuid(), getgrnam(), etc...`** are **not reentrant functions**
- ***SUSv3*** specifies an equivalent set of reentrant functions `getpwnam_r()`, `getpwuid_r()`, `getgrnam_r()`, and `getgrgid_r()` that include as arguments both **a passwd (or group) structure** and a **buffer** area to hold the other structures to which the fields of the passwd (group) structure point. The number of bytes required for this additional buffer can be obtained using the call `sysconf(_SC_GETPW_R_SIZE_MAX)` (or `sysconf(_SC_GETGR_R_SIZE_MAX)` in the case of the group-related functions). See the manual pages for details of these functions.
- According to ***SUSv3***, if a matching passwd record can't be found, then getpwnam() and getpwuid() should return NULL and leave errno unchanged. This means that we should be able to distinguish the error and the "not found" cases using code such as the following:

```c
int
main()
{
    struct passwd *ptr;
    errno = 0;
    const char *name = "lol";
    ptr = getpwnam(name);
    if(ptr == NULL){
        if(errno == 0){
            fprintf(stderr, "not found an entry to %s", name);
        }
        else{
            fprintf(stderr, "An error occured %s", strerror(errno));
        }

    }
    return 0;
}
```

- However, a number of UNIX implementations don't conform to ***SUSv3*** on this point. If a matching passwd record is not found, then these functions return NULL and set errno to a nonzero value, such as ENOENT or ESRCH. Before version 2.7, glibc produced the error ENOENT for this case, but since version 2.7, glibc conforms to the ***SUSv3*** requirements. This variation across implementations arises in part because POSIX.1-1990 did not require these functions to set errno on error and allowed them to set errno for the "not found" case. The upshot of all of this is that it isn't really possible to portably distinguish the error and "not found" cases when using these functions.
- **Retrieving records from the group file:**

```c
#include <grp.h>
struct group *getgrnam(const char *name);
struct group *getgrgid(gid_t gid);
// Both return a pointer on success, or NULL on error;
// see main text for description of the "not found" case
```

```
struct group {
    char  *gr_name;     /* Group name */
    char  *gr_passwd;   /* Encrypted password (if not password shadowing) */
    gid_t  gr_gid;      /* Group ID */
    char **gr_mem;      /* NULL-terminated array of pointers to names
                           of members listed in /etc/group */
};
```

- The ***gr_passwd*** field of the group structure is not specified in ***SUSv3***, but is available on most UNIX implementations.