【Linux Programming】Day5

Class	Understanding Linux/Unix Programming
Date	@March 17, 2022

【Ch2】Write who

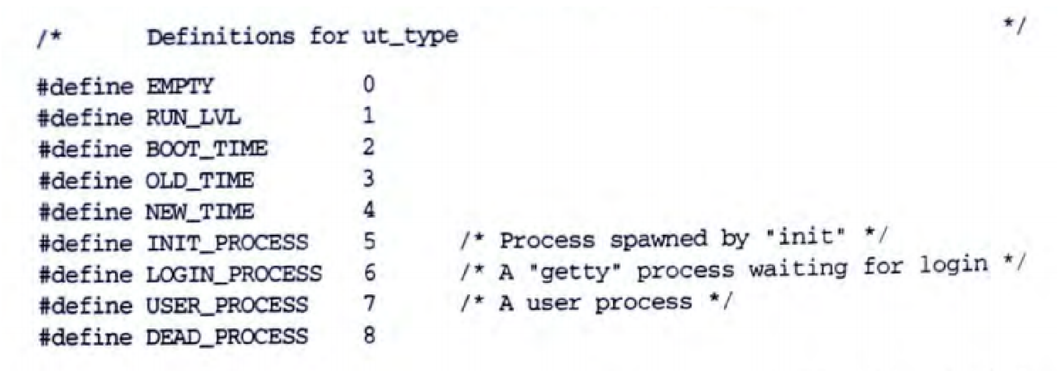
2.5.5 Writing who2.c

Suppressing Blank Records

Our version of `who` lists records for all terminals, even unused ones. The real version of `who` lists usernames of users logged into the system.

The solution is to print only the `utmp` records that represent users logged into the system.

Examining the `/usr/include/utmp.h` file, we find the following:



```
/*      Definitions for ut_type                                    */

#define EMPTY           0
#define RUN_LVL         1
#define BOOT_TIME       2
#define OLD_TIME        3
#define NEW_TIME        4
#define INIT_PROCESS    5       /* Process spawned by "init" */
#define LOGIN_PROCESS   6       /* A "getty" process waiting for login */
#define USER_PROCESS    7       /* A user process */
#define DEAD_PROCESS    8
```

Each record has a member called `ut_type`. The `USER_PROCESS` looks promising. We can thus make the following changes to our `who.c`

```
void show_info(struct utmp *record) {
  if(record->ut_type != USER_PROCESS)
    return;
}
```

Displaying Log-in Time in Human-Readable Form

Manpages on the topic of time vary a lot among versions of Unix.

We can type the following instructions to narrow the number of time results:

```
$ man -k time | grep transform
$ man -k time | grep -i convert
```

How Unix stores times: the time_t data type

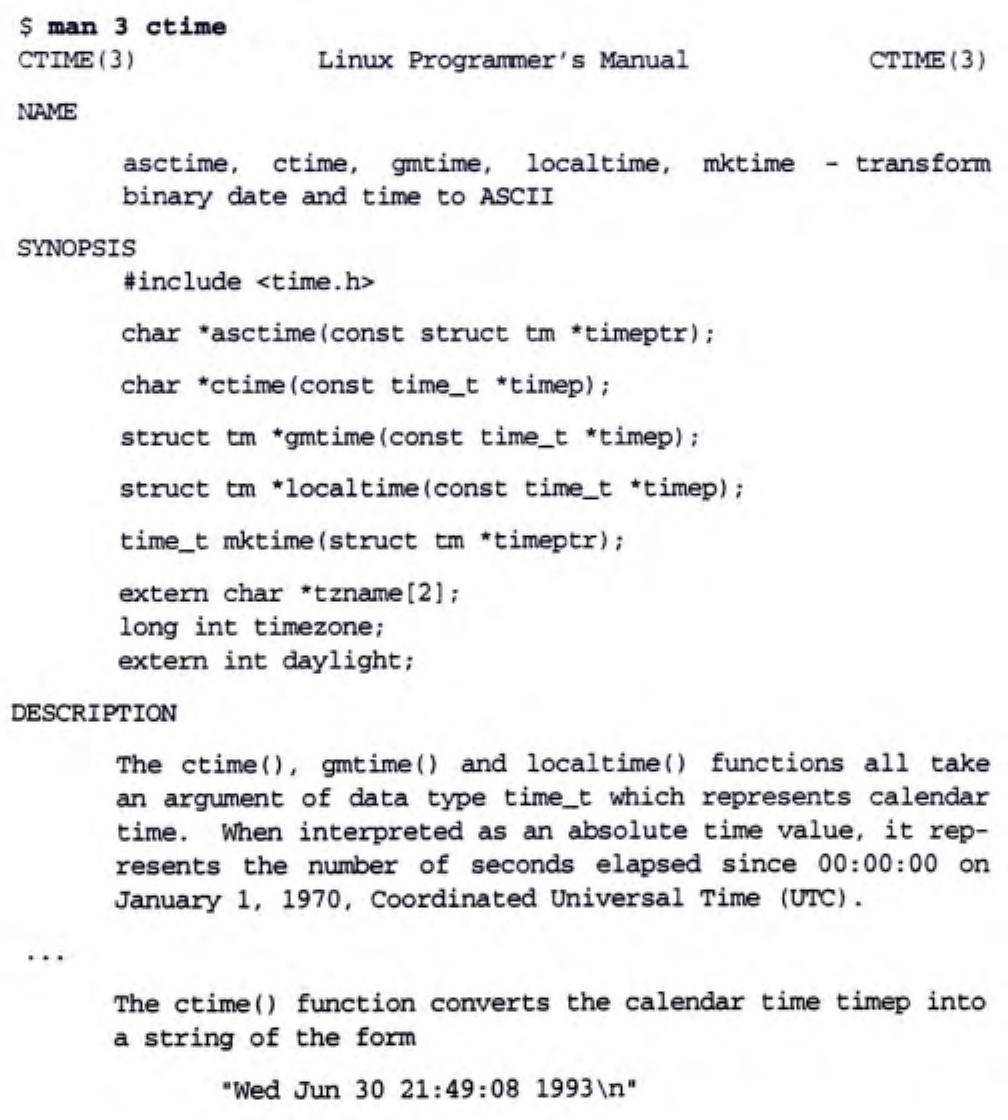
Unix stores times as the number of seconds since midnight, Jan 1, 1970, G.M.T.

The `time_t` data type is an integer that stores a number of seconds. Unix uses this format for many applications.

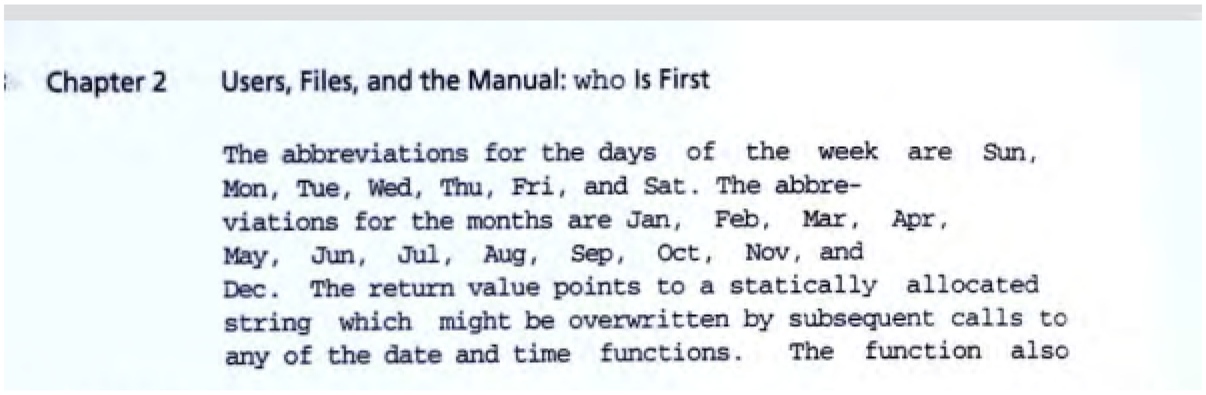
`ut_time` in the `utmp` records represents the log-in time as the number of seconds since the beginning of the Epoch.

Making a time_t readable: ctime

The function that converts a number of seconds since the start of Unix time into a sensible format is `ctime`, described in section 3 of the manual:



```
$ man 3 ctime
CTIME(3)            Linux Programmer's Manual            CTIME(3)

NAME
       asctime,  ctime,  gmtime,  localtime,  mktime  - transform
       binary date and time to ASCII

SYNOPSIS
       #include <time.h>

       char *asctime(const struct tm *timeptr);

       char *ctime(const time_t *timep);

       struct tm *gmtime(const time_t *timep);

       struct tm *localtime(const time_t *timep);

       time_t mktime(struct tm *timeptr);

       extern char *tzname[2];
       long int timezone;
       extern int daylight;

DESCRIPTION
       The ctime(), gmtime() and localtime() functions all take
       an argument of data type time_t which represents calendar
       time.  When interpreted as an absolute time value, it rep-
       resents the number of seconds elapsed since 00:00:00 on
       January 1, 1970, Coordinated Universal Time (UTC).

 ...

       The ctime() function converts the calendar time timep into
       a string of the form

              "Wed Jun 30 21:49:08 1993\n"
```



Chapter 2 Users, Files, and the Manual: who Is First

```
       The abbreviations for the days  of  the  week  are  Sun,
       Mon, Tue, Wed, Thu, Fri, and Sat. The abbre-
       viations for the months are Jan,  Feb,  Mar,  Apr,
       May,  Jun,  Jul,  Aug,  Sep,  Oct,  Nov, and
       Dec.  The return value points to a statically  allocated
       string  which  might be overwritten by subsequent calls to
       any of the date and time  functions.   The  function  also
```

Thus, we can call `ctime(utmp->ut_time)` to get the exact time and date.