산업과 예술의 만남
홍익대학교

Chapter 6.
System Data Files and Information

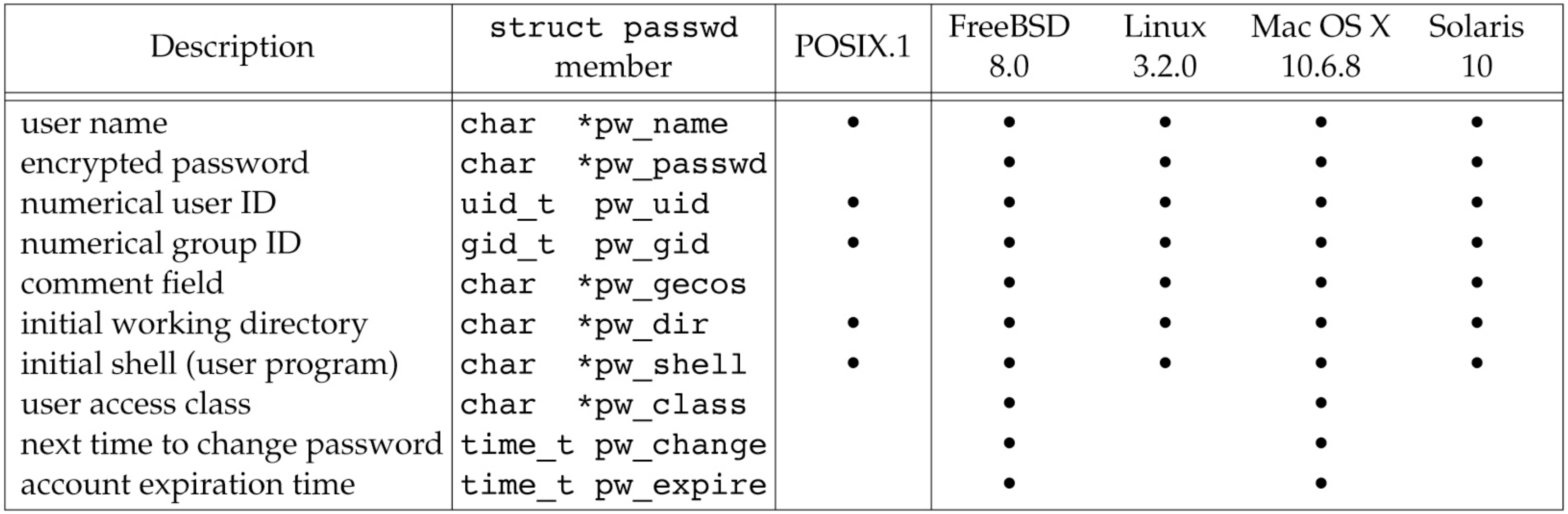
Hongik University

Eunsung Jung

Disclaimer: The slides are borrowed from many sources!

Password File

- Called *user database* in POSIX, and usually /etc/passwd
- Password file contains the following fields:



Description	struct passwd member	POSIX.1	FreeBSD 8.0	Linux 3.2.0	Mac OS X 10.6.8	Solaris 10
user name	`char *pw_name`	•	•	•	•	•
encrypted password	`char *pw_passwd`		•	•	•	•
numerical user ID	`uid_t pw_uid`	•	•	•	•	•
numerical group ID	`gid_t pw_gid`	•	•	•	•	•
comment field	`char *pw_gecos`		•	•	•	•
initial working directory	`char *pw_dir`	•	•	•	•	•
initial shell (user program)	`char *pw_shell`	•	•	•	•	•
user access class	`char *pw_class`		•		•	
next time to change password	`time_t pw_change`		•		•	
account expiration time	`time_t pw_expire`		•		•	

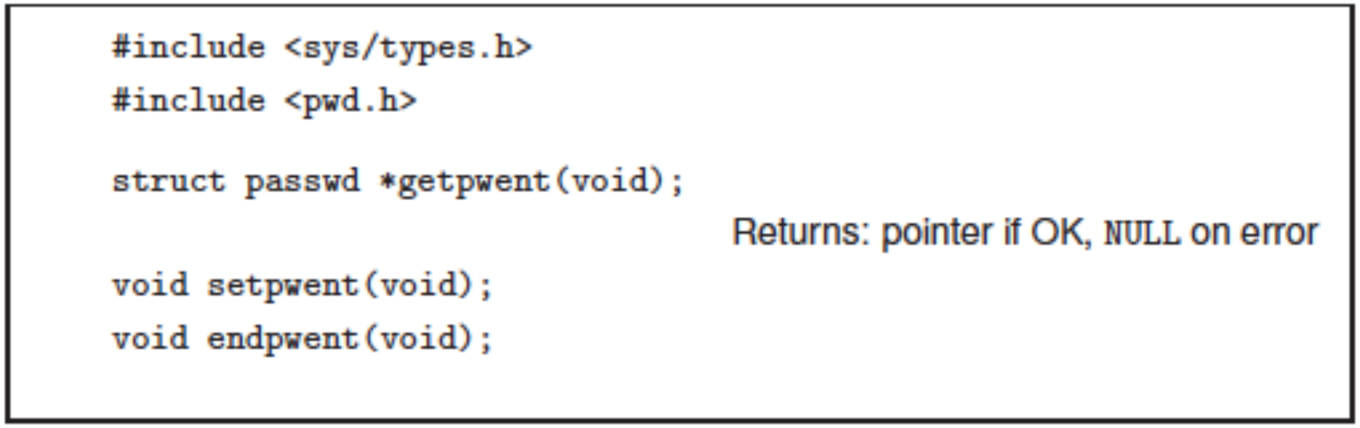
Figure 6.1 Fields in /etc/passwd file

Password File

- The encrypted password field contains a single character as a placeholder where older versions of the UNIX System used to store the encrypted password.
- Some fields can be empty:
  - password empty implies no password
  - shell empty implies /bin/sh
  - /dev/null: Nobody can log in as squid.

```
root:x:0:0:root:/root:/bin/bash
squid:x:23:23::/var/spool/squid:/dev/null
nobody:x:65534:65534:Nobody:/home:/bin/sh
sar:x:205:105:Stephen Rago:/home/sar:/bin/bash
```

Password File



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

struct passwd *getpwent(void);
                                        Returns: pointer if OK, NULL on error
void setpwent(void);
void endpwent(void);
```

- getpwent returns next password entry in file each time it's called, no order
- setpwent rewinds to ”beginning” of entries
- endpwent closes the file(s)

Password File

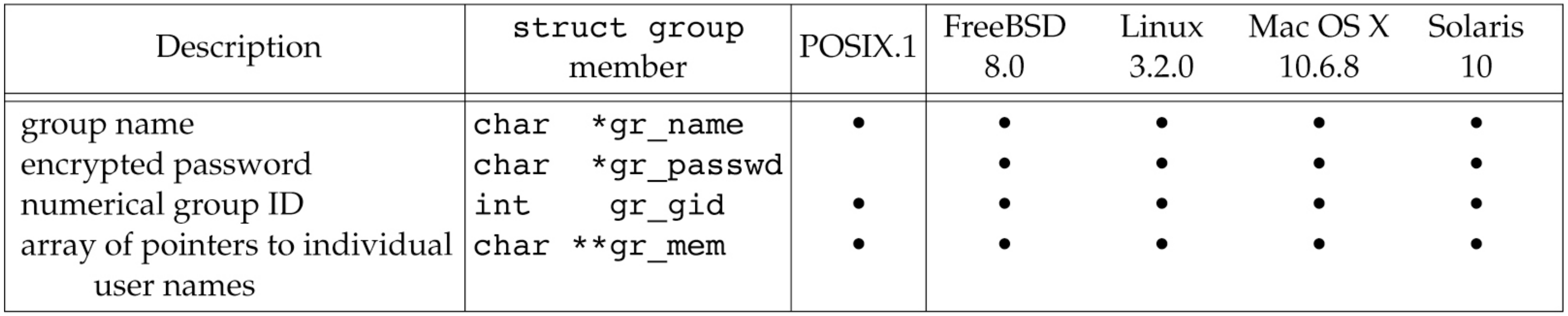
```
#include <sys/types.h>
#include <pwd.h>

struct passwd *getpwuid(uid_t uid);
struct passwd *getpwnam(const char *name);

                    Returns: pointer if OK, NULL on error
```

```
#include <pwd.h>
#include <stddef.h>
#include <string.h>

struct passwd *
getpwnam(const char *name)
{
    struct passwd  *ptr;

    setpwent();
    while ((ptr = getpwent()) != NULL)
        if (strcmp(name, ptr->pw_name) == 0)
            break;      /* found a match */
    endpwent();
    return(ptr);    /* ptr is NULL if no match found */
}
```

Figure 6.2 The getpwnam function

Group File

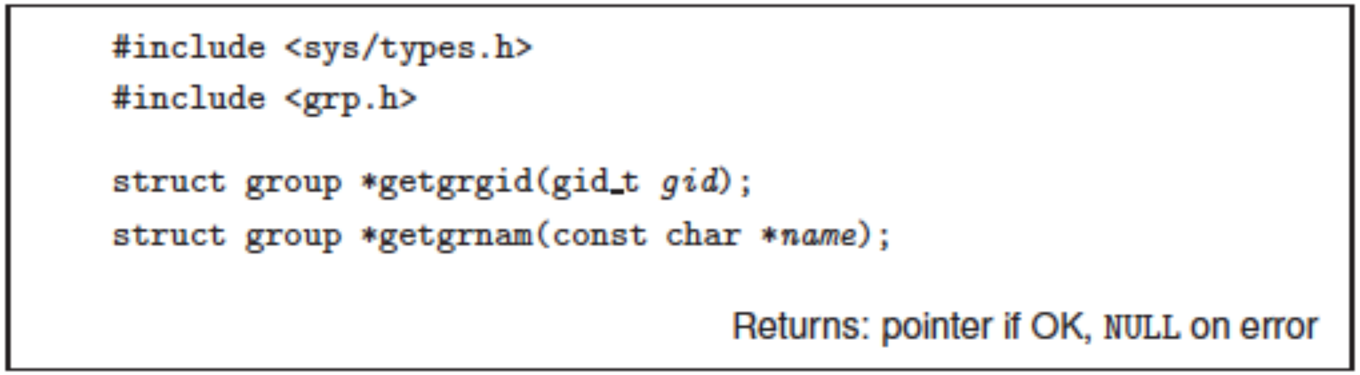
- Called *user database* in POSIX, and usually /etc/group
- Group file contains the following fields:



Description	`struct group` member	POSIX.1	FreeBSD 8.0	Linux 3.2.0	Mac OS X 10.6.8	Solaris 10
group name	`char *gr_name`	•	•	•	•	•
encrypted password	`char *gr_passwd`		•	•	•	•
numerical group ID	`int gr_gid`	•	•	•	•	•
array of pointers to individual user names	`char **gr_mem`	•	•	•	•	•

Figure 6.4 Fields in /etc/group file

Group File

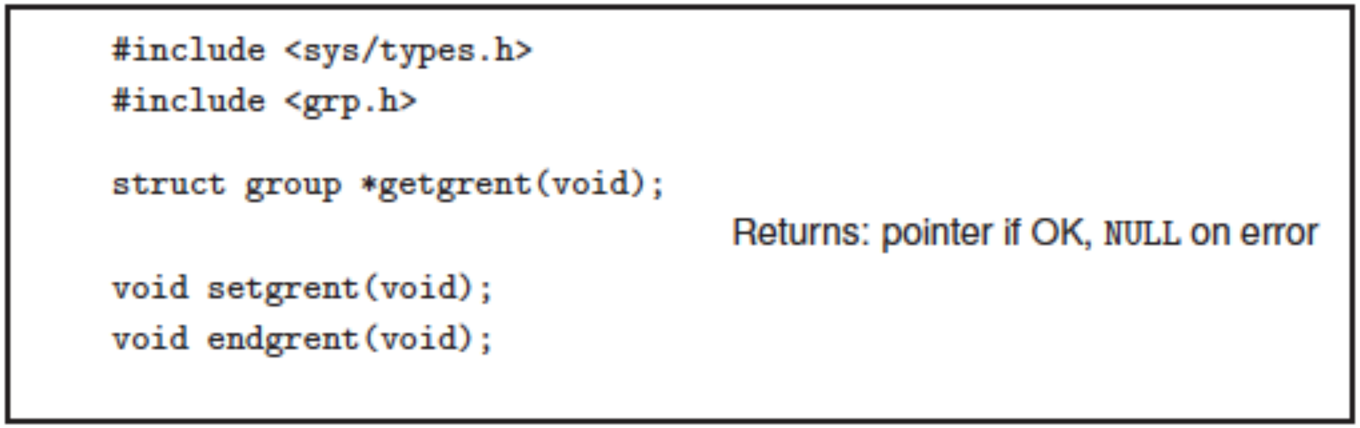


```
#include <sys/types.h>
#include <grp.h>

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

Returns: pointer if OK, NULL on error

- These allow us to look up an entry given a user’s group name or numerical GID.

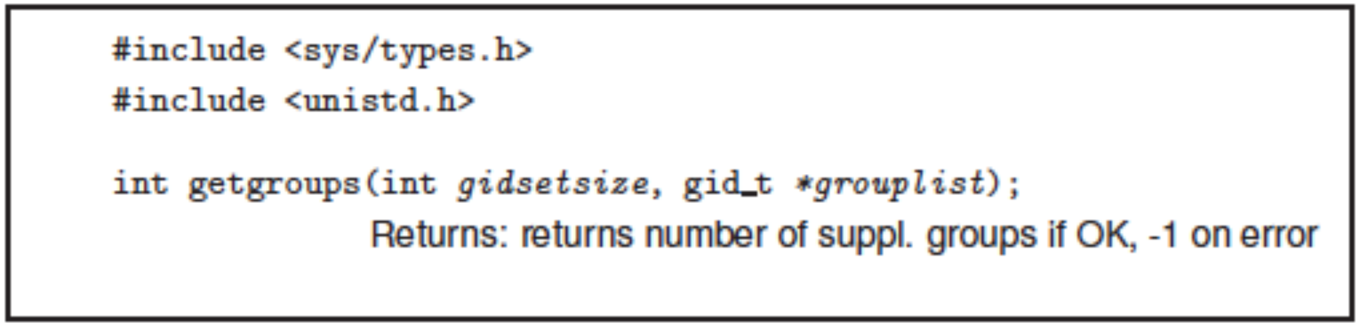
Group File



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

struct group *getgrent(void);
                                        Returns: pointer if OK, NULL on error

void setgrent(void);
void endgrent(void);
```

- What if we need to go through the group file entry by entry? Nothing in POSIX.1, but SVR4 and BSD give us:
- getgrent returns next group entry in file each time it’s called, no order.
- setgrent rewinds to ”beginning” of entries.
- endgrent closes the file(s).

Supplementary Groups and other data files



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

int getgroups(int gidsetsize, gid_t *grouplist);
```

Returns: returns number of suppl. groups if OK, -1 on error

- In old times, a user belonged to a single group at any point of time.
- But now multiple groups (supplementary group IDs) are possible. (POSIX.1)
- if gidsetsize == 0, getgroups(2) returns number of groups without modifying grouplist.

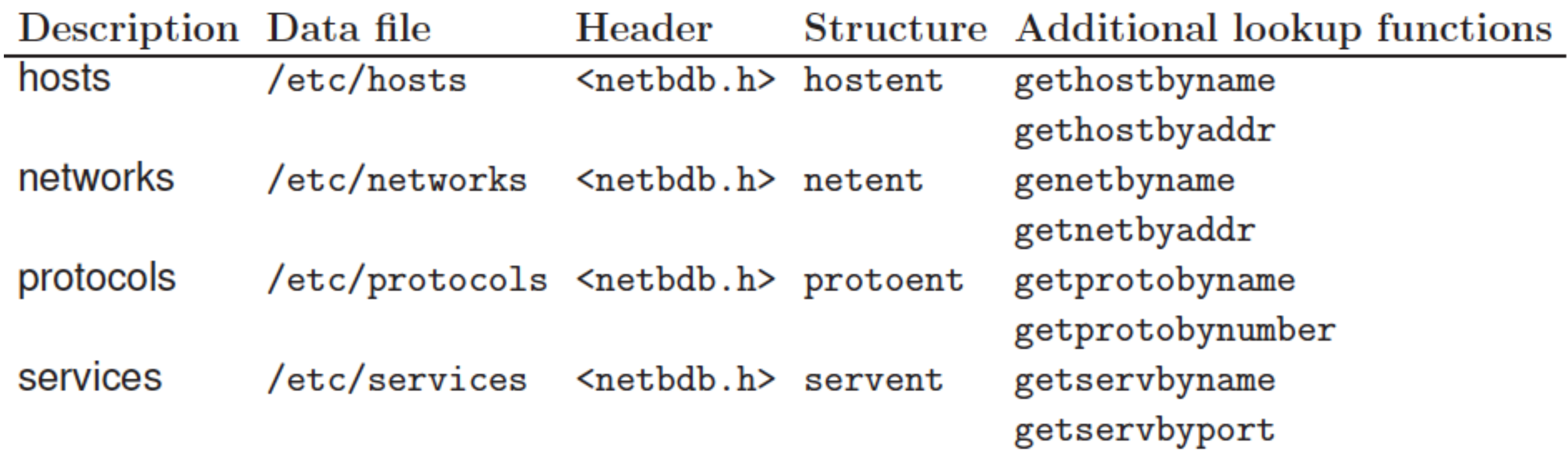
Account Implementation Differences

Information	FreeBSD 8.0	Linux 3.2.0	Mac OS X 10.6.8	Solaris 10
account information	`/etc/passwd`	`/etc/passwd`	Directory Services	`/etc/passwd`
encrypted passwords	`/etc/master.passwd`	`/etc/shadow`	Directory Services	`/etc/shadow`
hashed password files?	yes	no	no	no
group information	`/etc/group`	`/etc/group`	Directory Services	`/etc/group`

Figure 6.5 Account implementation differences

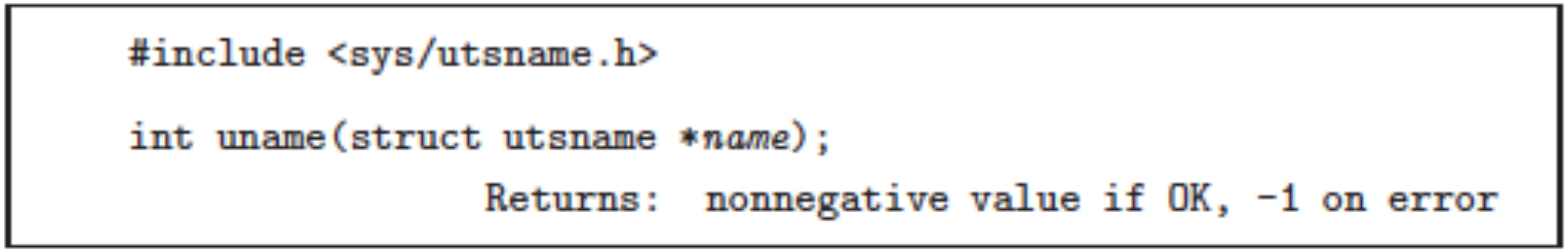
Other system databases

- Similar routines as for password/group for accessing system data files:



Description	Data file	Header	Structure	Additional lookup functions
hosts	/etc/hosts	<netbdb.h>	hostent	gethostbyname gethostbyaddr
networks	/etc/networks	<netbdb.h>	netent	genetbyname getnetbyaddr
protocols	/etc/protocols	<netbdb.h>	protoent	getprotobyname getprotobynumber
services	/etc/services	<netbdb.h>	servent	getservbyname getservbyport

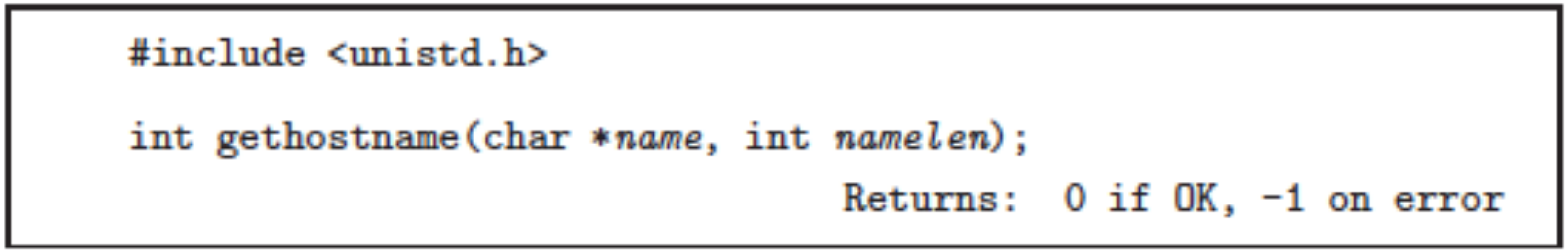
System Identification



```
#include <sys/utsname.h>

int uname(struct utsname *name);
                          Returns:  nonnegative value if OK, -1 on error
```

- Pass a pointer to a utsname struct. This struct contains fields like opsys name, version, release, architecture, etc.
- This function used by the uname(1) command (try uname -a)
- Not that the size of the fields in the utsname struct may not be large enough to id a host on a network

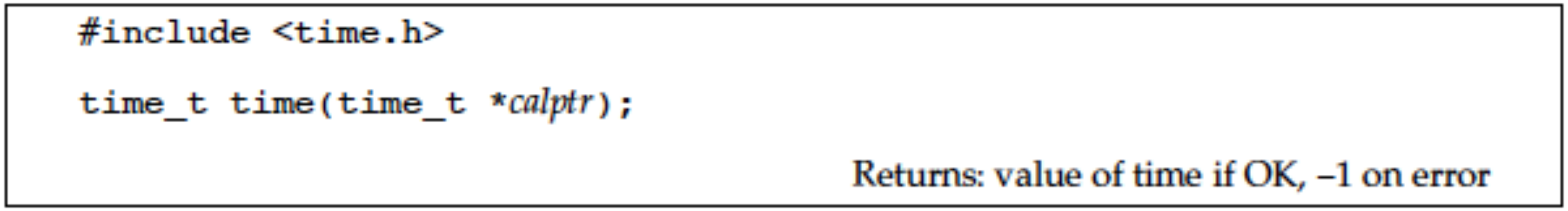
System Identification



```
#include <unistd.h>

int gethostname(char *name, int namelen);
                                        Returns: 0 if OK, -1 on error
```

- To get just a hostname that will identify you on a TCP/IP network, use the Berkeley-dervied.
- try hostname

Time and Date



```
#include <time.h>

time_t time(time_t *calptr);
```

Returns: value of time if OK, –1 on error

- Time is kept in UTC.
  - the number of seconds that have passed since the Epoch: 00:00:00 January 1, 1970, Coordinated Universal Time(UTC).
- Time conversions (timezone, daylight savings time) handled ”automatically”
- Time and date kept in a single quantity (time_t)

Time and Date

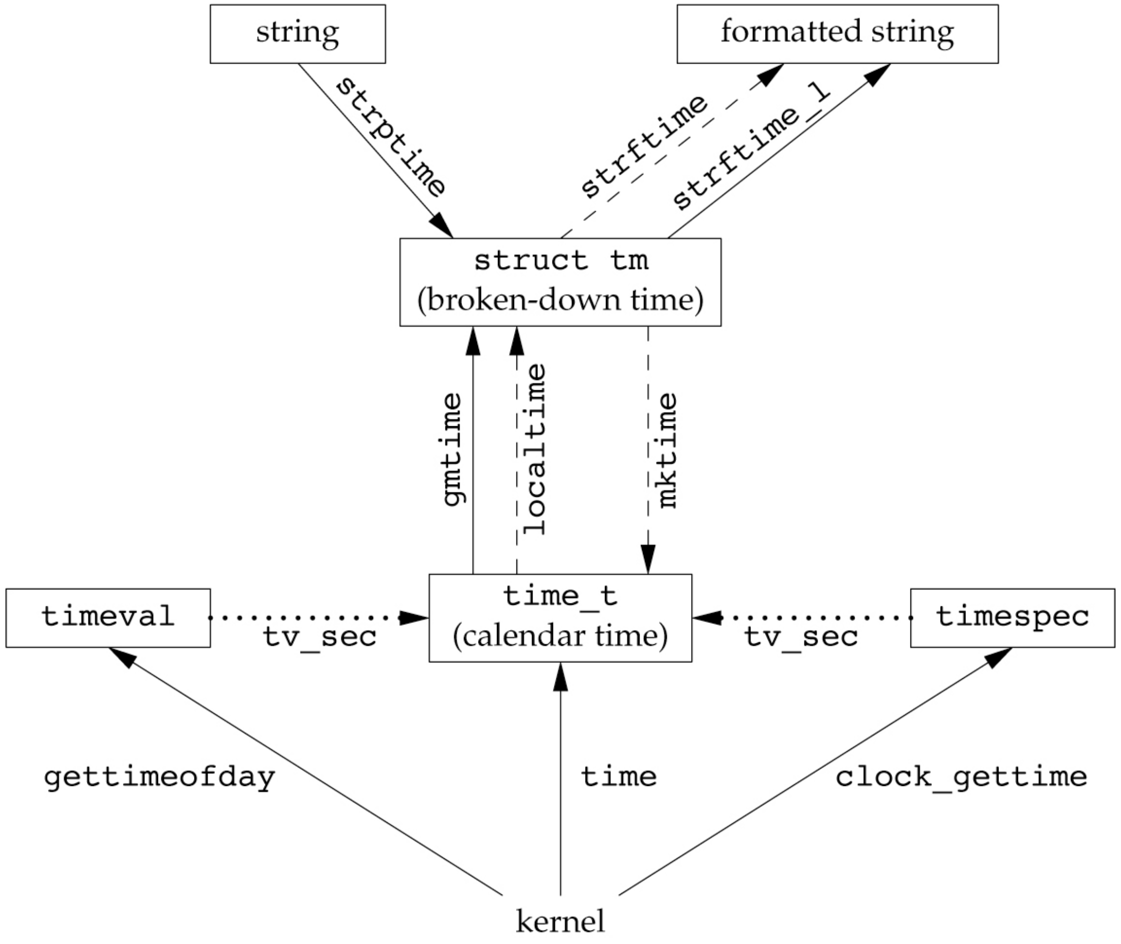
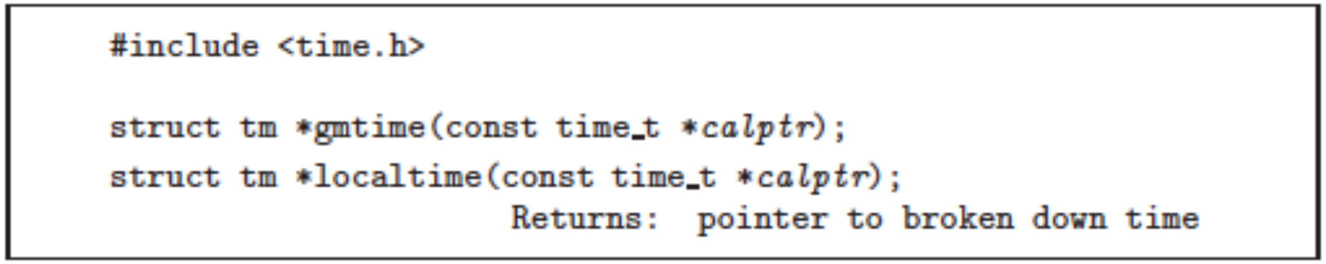


Figure 6.9 Relationship of the various time functions

Time and Date

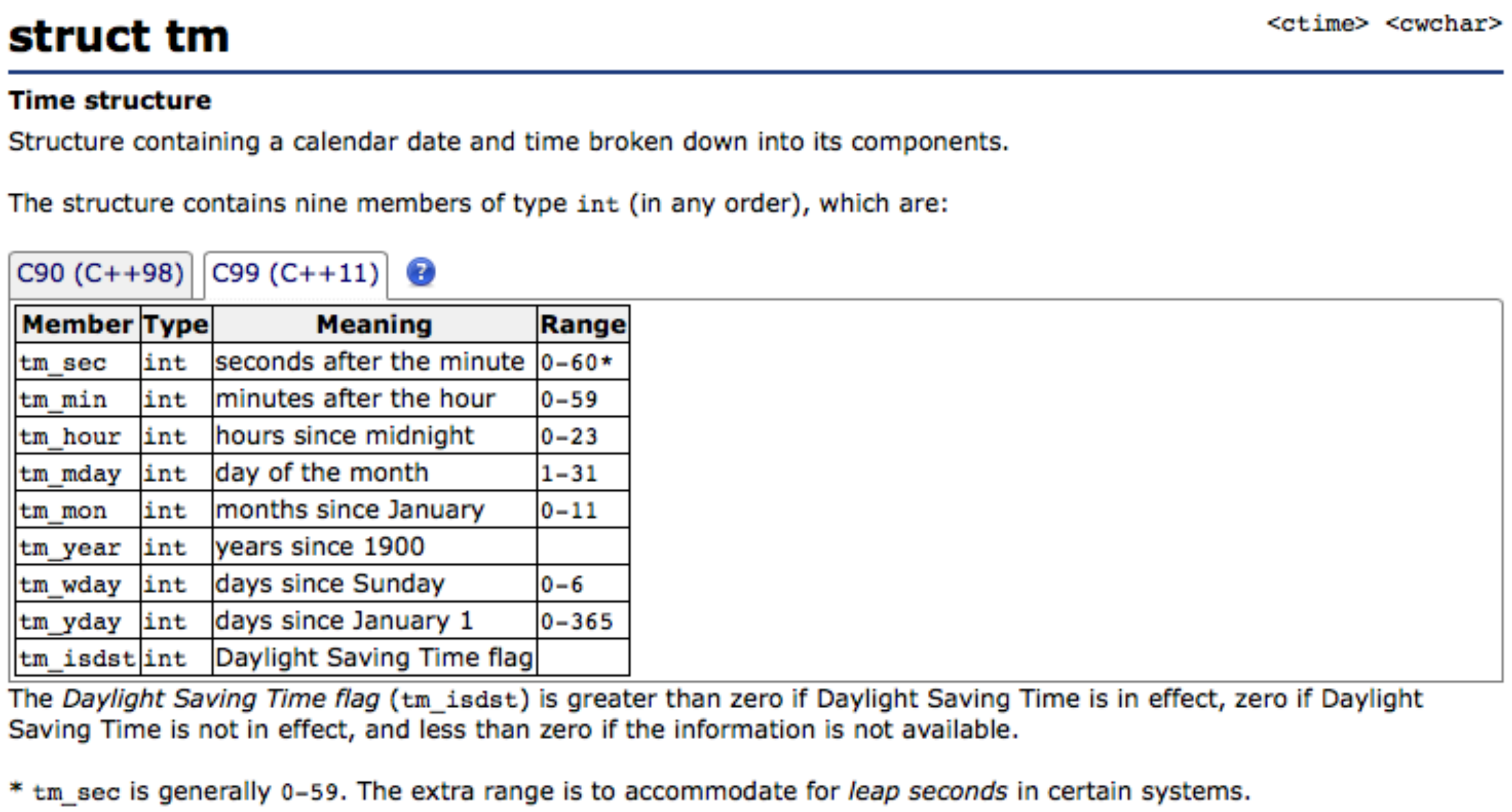
- We can break this time_t value into its components with either of the following:



```
#include <time.h>

struct tm *gmtime(const time_t *calptr);
struct tm *localtime(const time_t *calptr);
                       Returns: pointer to broken down time
```

- localtime(3) takes into account daylight savings time and the TZ environment variable.

Time and Date



struct tm

`<ctime> <cwchar>`

Time structure

Structure containing a calendar date and time broken down into its components.

The structure contains nine members of type `int` (in any order), which are:

C90 (C++98) | C99 (C++11)

Member	Type	Meaning	Range
`tm_sec`	`int`	seconds after the minute	`0-60*`
`tm_min`	`int`	minutes after the hour	`0-59`
`tm_hour`	`int`	hours since midnight	`0-23`
`tm_mday`	`int`	day of the month	`1-31`
`tm_mon`	`int`	months since January	`0-11`
`tm_year`	`int`	years since 1900	
`tm_wday`	`int`	days since Sunday	`0-6`
`tm_yday`	`int`	days since January 1	`0-365`
`tm_isdst`	`int`	Daylight Saving Time flag	

The *Daylight Saving Time flag* (`tm_isdst`) is greater than zero if Daylight Saving Time is in effect, zero if Daylight Saving Time is not in effect, and less than zero if the information is not available.

* `tm_sec` is generally `0-59`. The extra range is to accommodate for *leap seconds* in certain systems.

Time and Date

- To output human readable results, use:

```
#include <time.h>

char *asctime(const struct tm *tmptr);
char *ctime(const struct tm *tmptr);
                      Returns:  pointer to NULL terminated string
```

- Lastly, there is a printf(3) like function for times:

```
#include <time.h>

size_t strftime(char *buf, size_t maxsize, const char *restricted format, const struct tm *timeptr);
                                   Returns:  number of characters stored in array if room, else 0
```

Lab: Exercise 6.3

- Write a program that calls uname and prints all the fields in the utsname structure. Compare the output to the output from the uname(1) command.