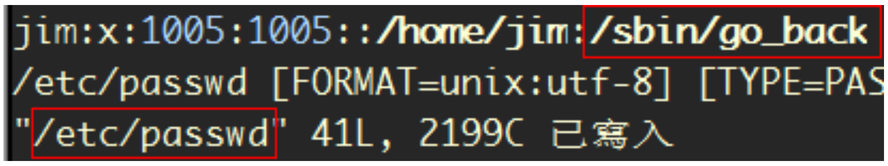
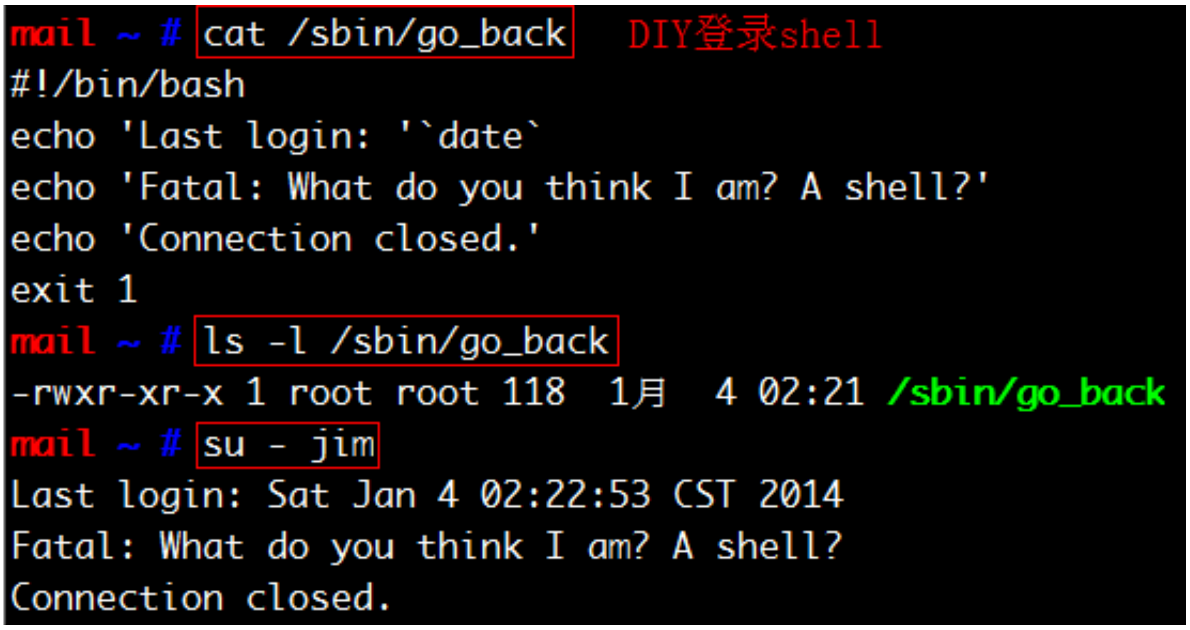
第六章 系统数据文件和信息

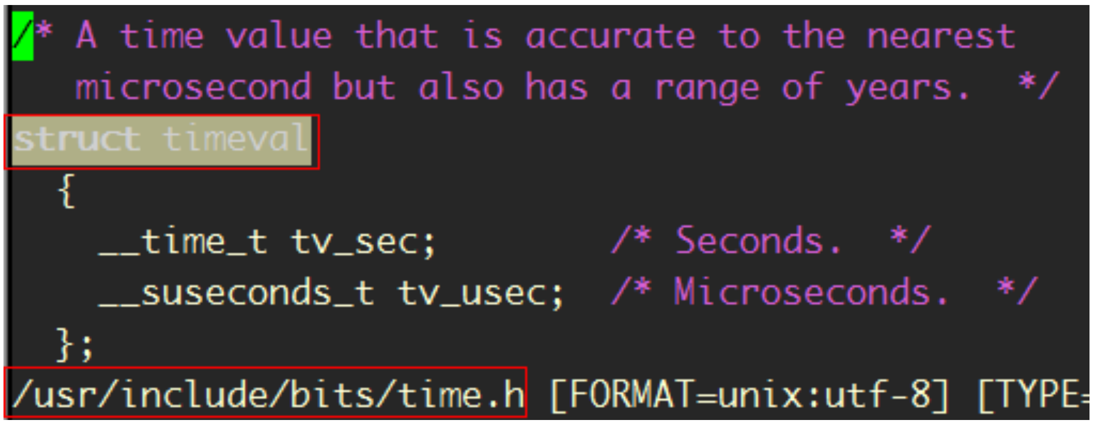


```
jim:x:1005:1005::/home/jim:/sbin/go_back
/etc/passwd [FORMAT=unix:utf-8] [TYPE=PAS
"/etc/passwd" 41L, 2199C 已寫入
```

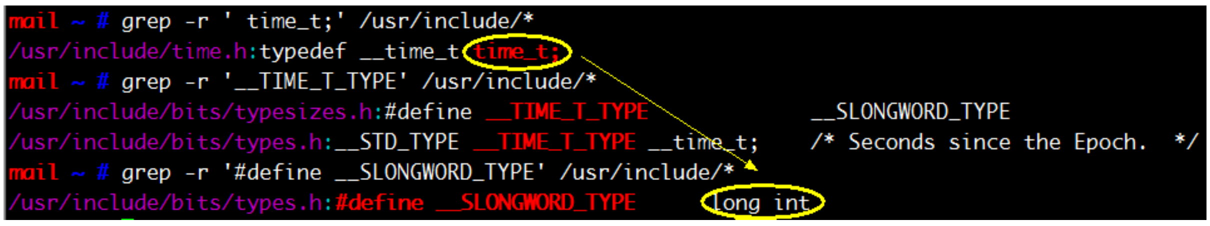


```
mail ~ # cat /sbin/go_back    DIY登录shell
#!/bin/bash
echo 'Last login: '`date`
echo 'Fatal: What do you think I am? A shell?'
echo 'Connection closed.'
exit 1
mail ~ # ls -l /sbin/go_back
-rwxr-xr-x 1 root root 118  1月  4 02:21 /sbin/go_back
mail ~ # su - jim
Last login: Sat Jan 4 02:22:53 CST 2014
Fatal: What do you think I am? A shell?
Connection closed.
```

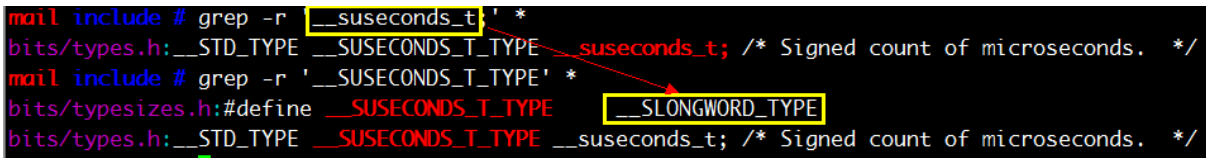
是不是很拉风呀。



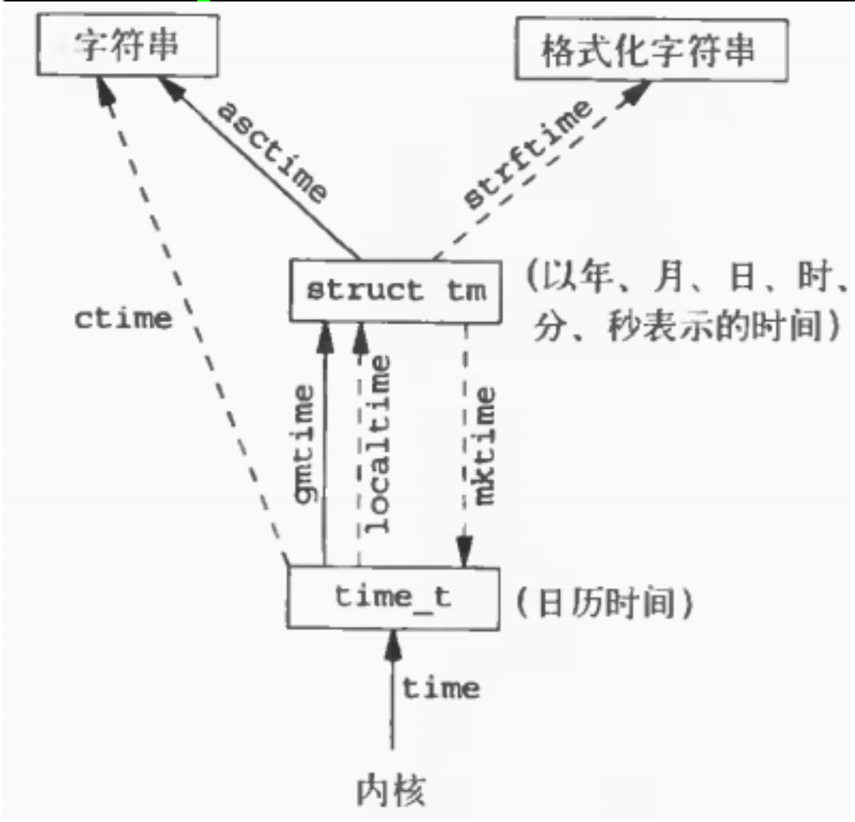
```
/* A time value that is accurate to the nearest
   microsecond but also has a range of years.  */
struct timeval
  {
    __time_t tv_sec;          /* Seconds.  */
    __suseconds_t tv_usec;    /* Microseconds.  */
  };
/usr/include/bits/time.h [FORMAT=unix:utf-8] [TYPE=
```

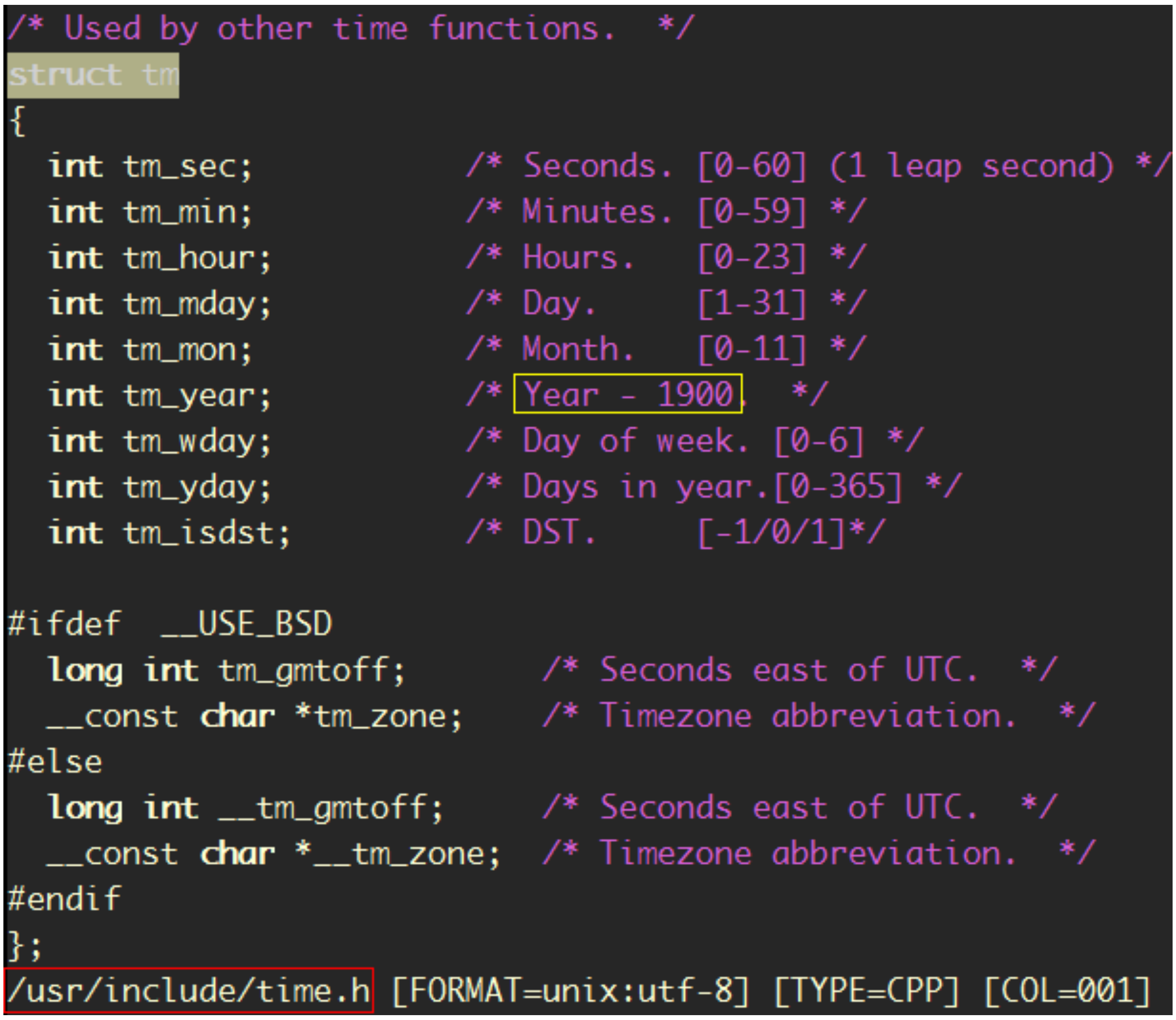


```
mail ~ # grep -r ' time_t;' /usr/include/*
/usr/include/time.h:typedef __time_t time_t;
mail ~ # grep -r '__TIME_T_TYPE' /usr/include/*
/usr/include/bits/typesizes.h:#define __TIME_T_TYPE          __SLONGWORD_TYPE
/usr/include/bits/types.h:__STD_TYPE __TIME_T_TYPE __time_t;    /* Seconds since the Epoch.  */
mail ~ # grep -r '#define __SLONGWORD_TYPE' /usr/include/*
/usr/include/bits/types.h:#define __SLONGWORD_TYPE      long int
```



```
mail include # grep -r '__suseconds_t;' *
bits/types.h:__STD_TYPE __SUSECONDS_T_TYPE __suseconds_t; /* Signed count of microseconds.  */
mail include # grep -r '__SUSECONDS_T_TYPE' *
bits/typesizes.h:#define __SUSECONDS_T_TYPE      __SLONGWORD_TYPE
bits/types.h:__STD_TYPE __SUSECONDS_T_TYPE __suseconds_t; /* Signed count of microseconds.  */
```

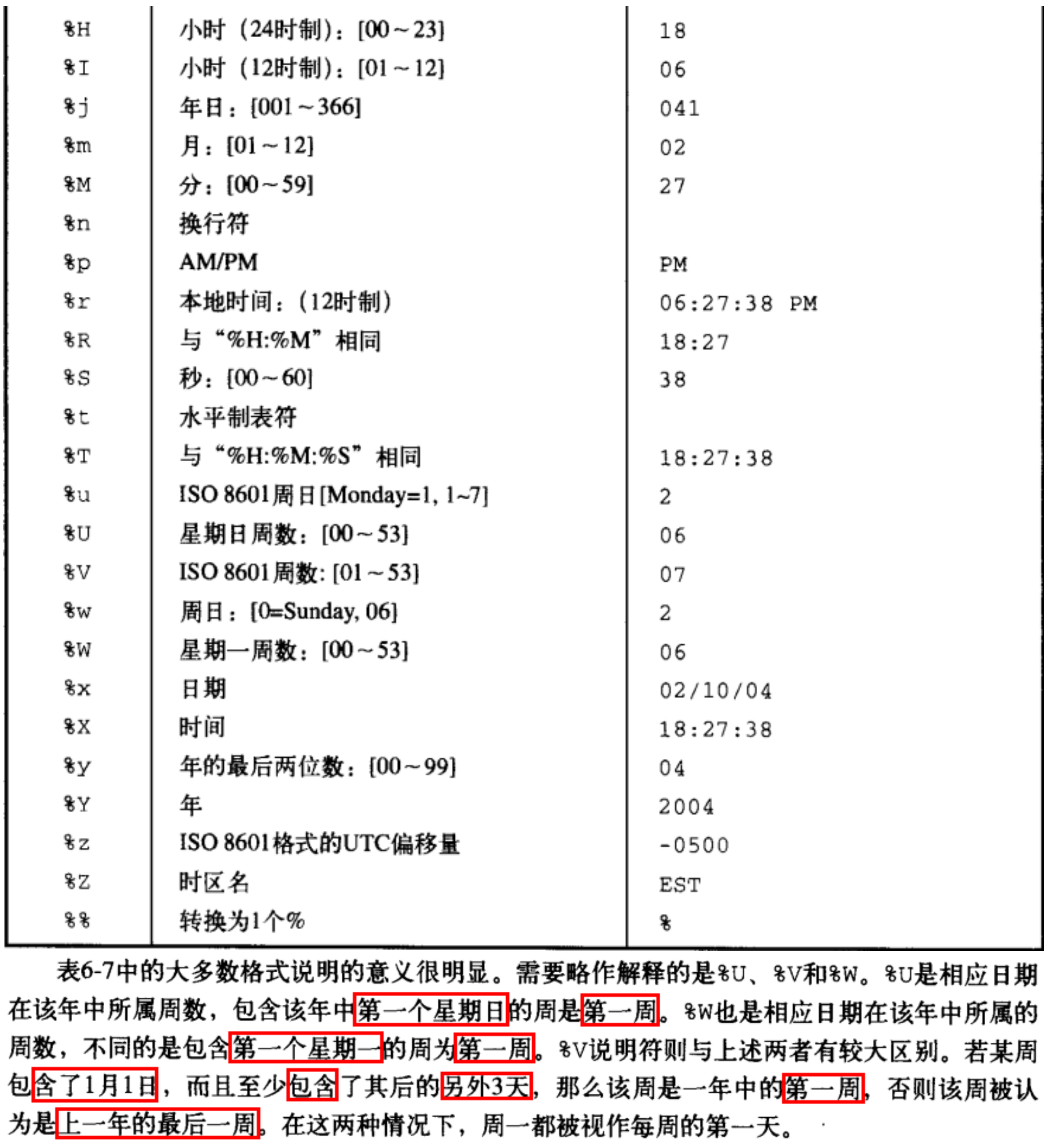




```
/* Used by other time functions.  */
struct tm
{
  int tm_sec;                   /* Seconds. [0-60] (1 leap second) */
  int tm_min;                   /* Minutes. [0-59] */
  int tm_hour;                  /* Hours.   [0-23] */
  int tm_mday;                  /* Day.     [1-31] */
  int tm_mon;                   /* Month.   [0-11] */
  int tm_year;                  /* Year - 1900.  */
  int tm_wday;                  /* Day of week. [0-6] */
  int tm_yday;                  /* Days in year.[0-365] */
  int tm_isdst;                 /* DST.     [-1/0/1]*/

#ifdef  __USE_BSD
  long int tm_gmtoff;           /* Seconds east of UTC.  */
  __const char *tm_zone;        /* Timezone abbreviation.  */
#else
  long int __tm_gmtoff;         /* Seconds east of UTC.  */
  __const char *__tm_zone;      /* Timezone abbreviation.  */
#endif
};
/usr/include/time.h [FORMAT=unix:utf-8] [TYPE=CPP] [COL=001]
```

表6-7 strftime的转换说明

格　式	说　　明	实　例
%a	缩写的周日名	Tue
%A	全周日名	Tuesday
%b	缩写的月名	Feb
%B	全月名	February
%c	日期和时间	Tue Feb 10 18:27:38 2004
%C	年/100：[00～99]	20
%d	月日：[01～31]	10
%D	日期 [MM/DD/YY]	02/10/04
%e	月日（一位数前加空格）：[1～31]	10
%F	ISO 8601 日期格式 [YYYY-MM-DD]	2004-02-10
%g	ISO 8601基于周的年的最后2位数[00~99]	04
%G	ISO 8601基于周的年	2004
%h	与%b相同	Feb

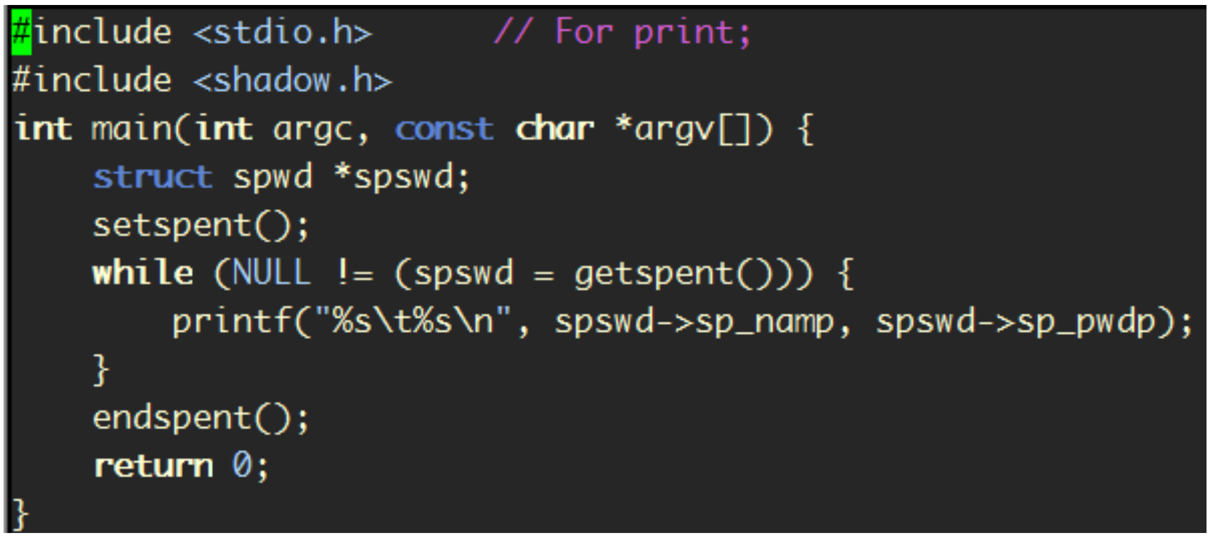


%H	小时（24时制）：[00～23]	18
%I	小时（12时制）：[01～12]	06
%j	年日：[001～366]	041
%m	月：[01～12]	02
%M	分：[00～59]	27
%n	换行符	
%p	AM/PM	PM
%r	本地时间：（12时制）	06:27:38 PM
%R	与“%H:%M”相同	18:27
%S	秒：[00～60]	38
%t	水平制表符	
%T	与“%H:%M:%S”相同	18:27:38
%u	ISO 8601周日[Monday=1, 1~7]	2
%U	星期日周数：[00～53]	06
%V	ISO 8601周数: [01～53]	07
%w	周日：[0=Sunday, 06]	2
%W	星期一周数：[00～53]	06
%x	日期	02/10/04
%X	时间	18:27:38
%y	年的最后两位数：[00～99]	04
%Y	年	2004
%z	ISO 8601格式的UTC偏移量	-0500
%Z	时区名	EST
%%	转换为1个%	%

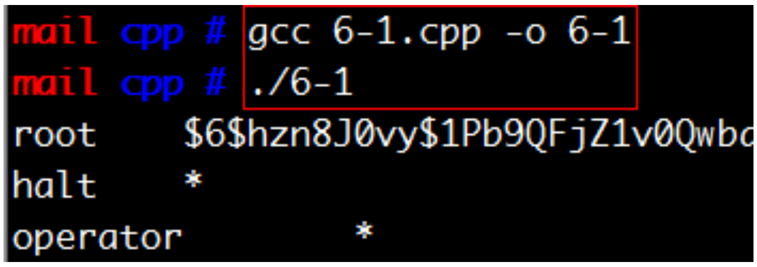
表6-7中的大多数格式说明的意义很明显。需要略作解释的是%U、%V和%W。%U是相应日期在该年中所属周数，包含该年中第一个星期日的周是第一周。%W也是相应日期在该年中所属的周数，不同的是包含第一个星期一的周为第一周。%V说明符则与上述两者有较大区别。若某周包含了1月1日，而且至少包含了其后的另外3天，那么该周是一年中的第一周，否则该周被认为是上一年的最后一周。在这两种情况下，周一都被视作每周的第一天。

习题：

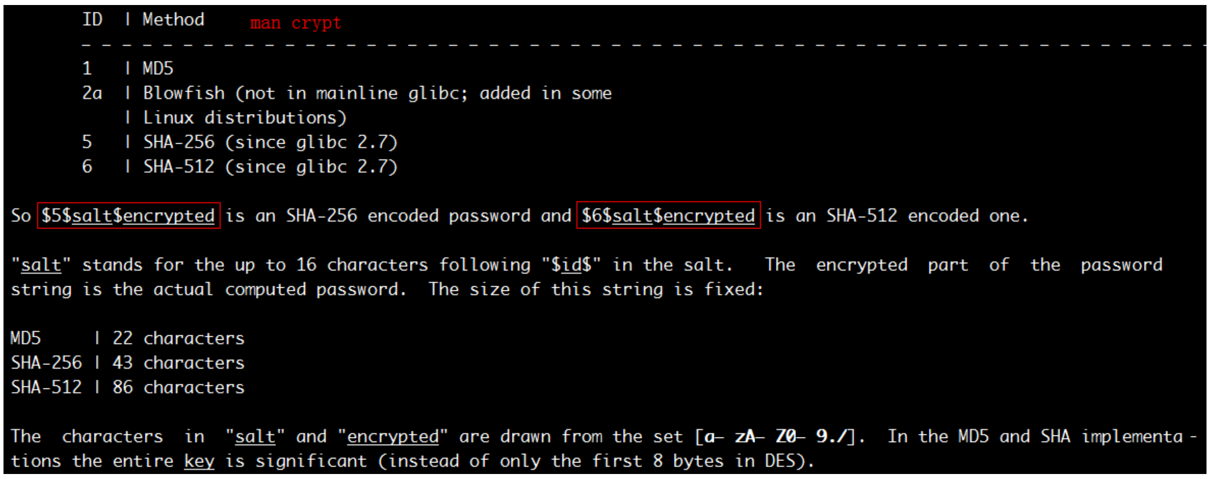
6-1 、easy；



```c
#include <stdio.h>        // For print;
#include <shadow.h>
int main(int argc, const char *argv[]) {
    struct spwd *spswd;
    setspent();
    while (NULL != (spswd = getspent())) {
        printf("%s\t%s\n", spswd->sp_namp, spswd->sp_pwdp);
    }
    endspent();
    return 0;
}
```

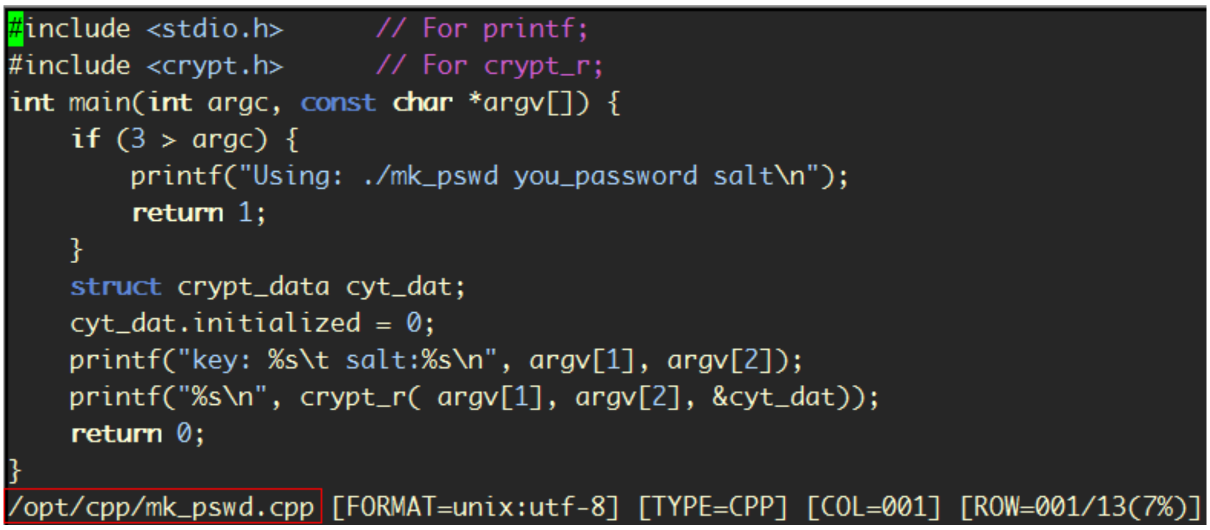


```
mail cpp # gcc 6-1.cpp -o 6-1
mail cpp # ./6-1
root     $6$hzn8J0vy$1Pb9QFjZ1v0Qwb
halt     *
operator         *
```

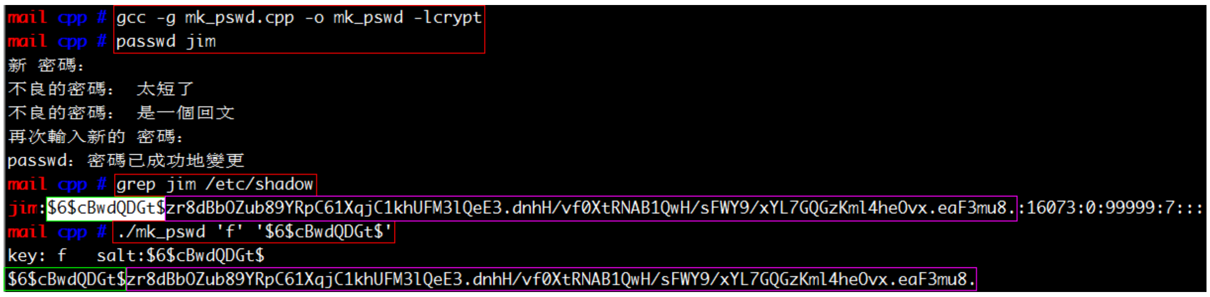


```
       ID  | Method      man crypt
       ─────────────────────────────────────────────────────────────────
       1   | MD5
       2a  | Blowfish (not in mainline glibc; added in some
           | Linux distributions)
       5   | SHA-256 (since glibc 2.7)
       6   | SHA-512 (since glibc 2.7)

So $5$salt$encrypted is an SHA-256 encoded password and $6$salt$encrypted is an SHA-512 encoded one.

"salt" stands for the up to 16 characters following "$id$" in the salt.   The  encrypted  part  of  the  password
string is the actual computed password.  The size of this string is fixed:

MD5     | 22 characters
SHA-256 | 43 characters
SHA-512 | 86 characters

The  characters  in  "salt" and "encrypted" are drawn from the set [a- zA- Z0- 9./].  In the MD5 and SHA implementa-
tions the entire key is significant (instead of only the first 8 bytes in DES).
```

6-1E：生成系统密码；



```
#include <stdio.h>       // For printf;
#include <crypt.h>       // For crypt_r;
int main(int argc, const char *argv[]) {
    if (3 > argc) {
        printf("Using: ./mk_pswd you_password salt\n");
        return 1;
    }
    struct crypt_data cyt_dat;
    cyt_dat.initialized = 0;
    printf("key: %s\t salt:%s\n", argv[1], argv[2]);
    printf("%s\n", crypt_r( argv[1], argv[2], &cyt_dat));
    return 0;
}
/opt/cpp/mk_pswd.cpp [FORMAT=unix:utf-8] [TYPE=CPP] [COL=001] [ROW=001/13(7%)]
```

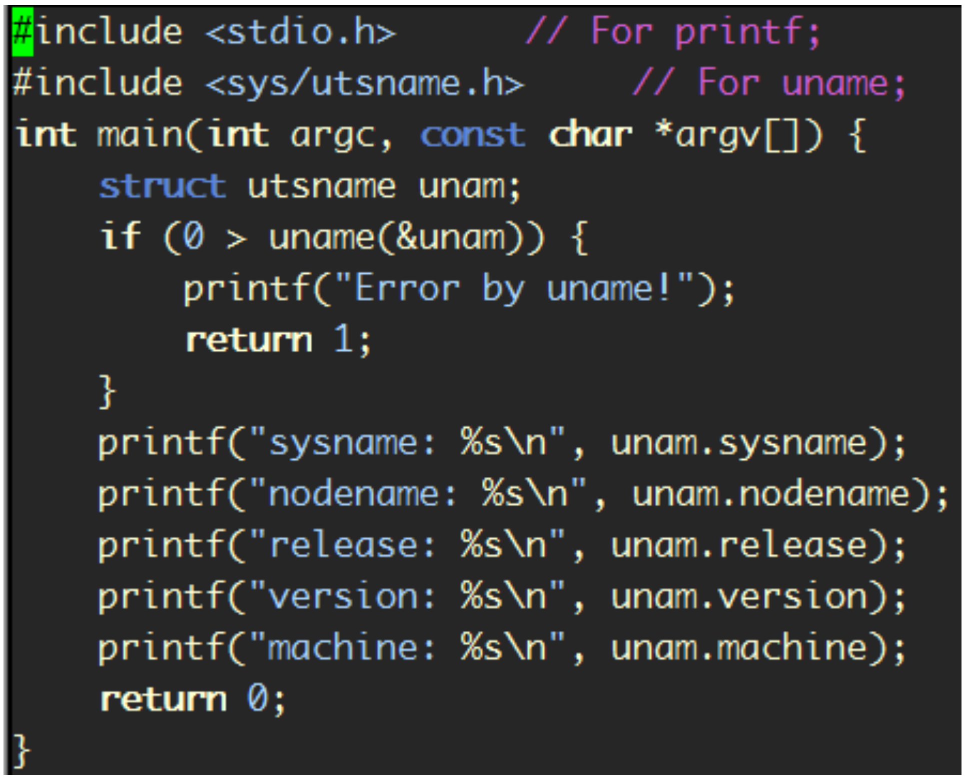


```
mail cpp # gcc -g mk_pswd.cpp -o mk_pswd -lcrypt
mail cpp # passwd jim
新 密碼:
不良的密碼:  太短了
不良的密碼:  是一個回文
再次輸入新的 密碼:
passwd: 密碼已成功地變更
mail cpp # grep jim /etc/shadow
jim:$6$cBwdQDGt$zr8dBbOZub89YRpC61XqjC1khUFM3lQeE3.dnhH/vf0XtRNAB1QwH/sFWY9/xYL7GQGzKml4heOvx.eaF3mu8.:16073:0:99999:7:::
mail cpp # ./mk_pswd 'f' '$6$cBwdQDGt$'
key: f   salt:$6$cBwdQDGt$
$6$cBwdQDGt$zr8dBbOZub89YRpC61XqjC1khUFM3lQeE3.dnhH/vf0XtRNAB1QwH/sFWY9/xYL7GQGzKml4heOvx.eaF3mu8.
```

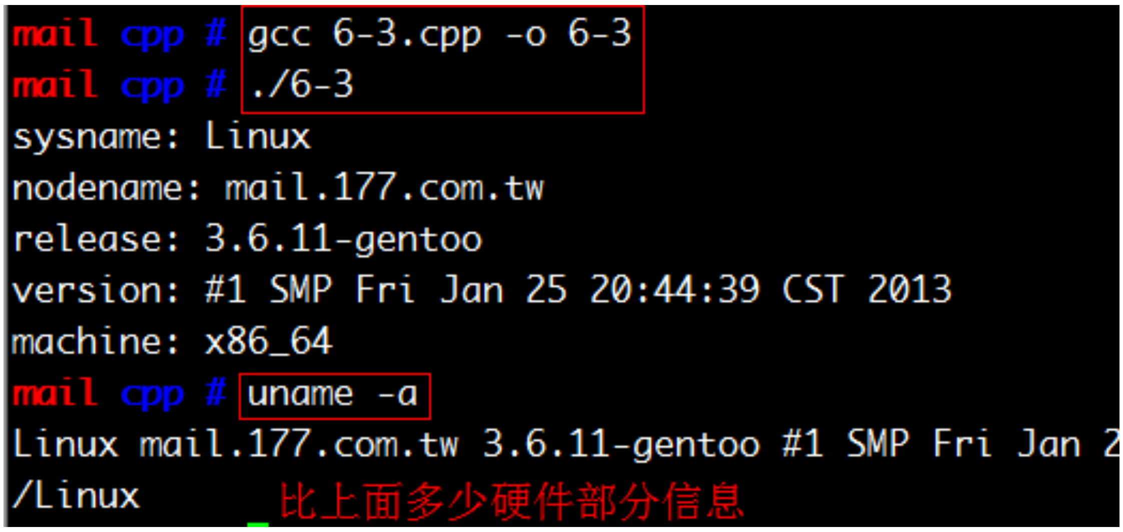
6-2、普通用户无法取得shadow内容，超级用户如上演示；

6-3、系统uname会输出少许硬件部分信息；

系统uname源码参考http://src.gnu-darwin.org/src/usr.bin/uname/uname.c.html

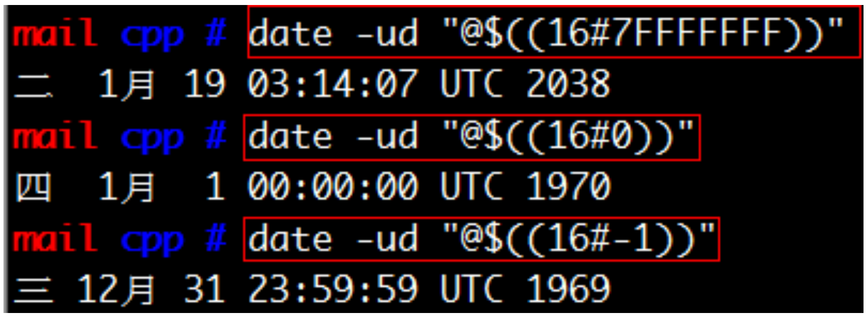


```c
#include <stdio.h>          // For printf;
#include <sys/utsname.h>       // For uname;
int main(int argc, const char *argv[]) {
    struct utsname unam;
    if (0 > uname(&unam)) {
        printf("Error by uname!");
        return 1;
    }
    printf("sysname: %s\n", unam.sysname);
    printf("nodename: %s\n", unam.nodename);
    printf("release: %s\n", unam.release);
    printf("version: %s\n", unam.version);
    printf("machine: %s\n", unam.machine);
    return 0;
}
```



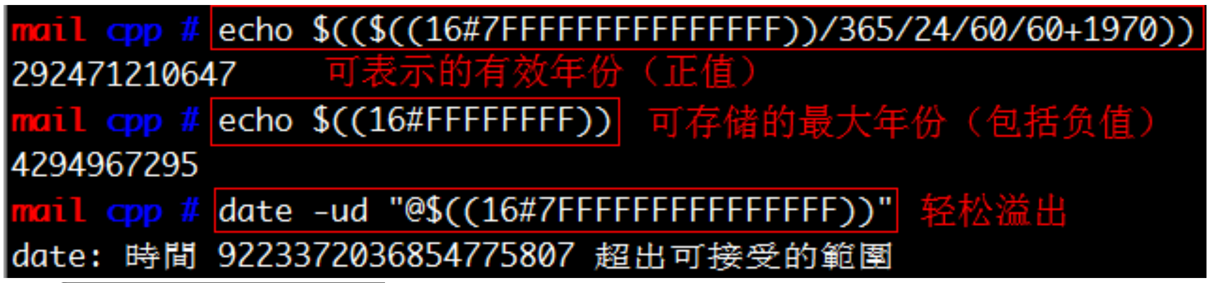
```
mail cpp # gcc 6-3.cpp -o 6-3
mail cpp # ./6-3
sysname: Linux
nodename: mail.177.com.tw
release: 3.6.11-gentoo
version: #1 SMP Fri Jan 25 20:44:39 CST 2013
machine: x86_64
mail cpp # uname -a
Linux mail.177.com.tw 3.6.11-gentoo #1 SMP Fri Jan 2
/Linux
```

6-4、分两种情况，time_t为32位或64位；
当为32位有符号整形时，如下；

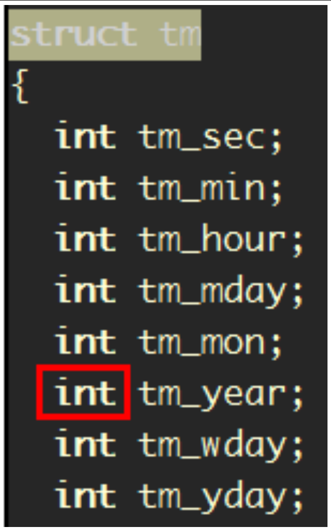


```
mail cpp # date -ud "@$((16#7FFFFFFF))"
二  1月  19 03:14:07 UTC 2038
mail cpp # date -ud "@$((16#0))"
四  1月   1 00:00:00 UTC 1970
mail cpp # date -ud "@$((16#-1))"
三 12月  31 23:59:59 UTC 1969
```

当为64为有符号整形时，如下；



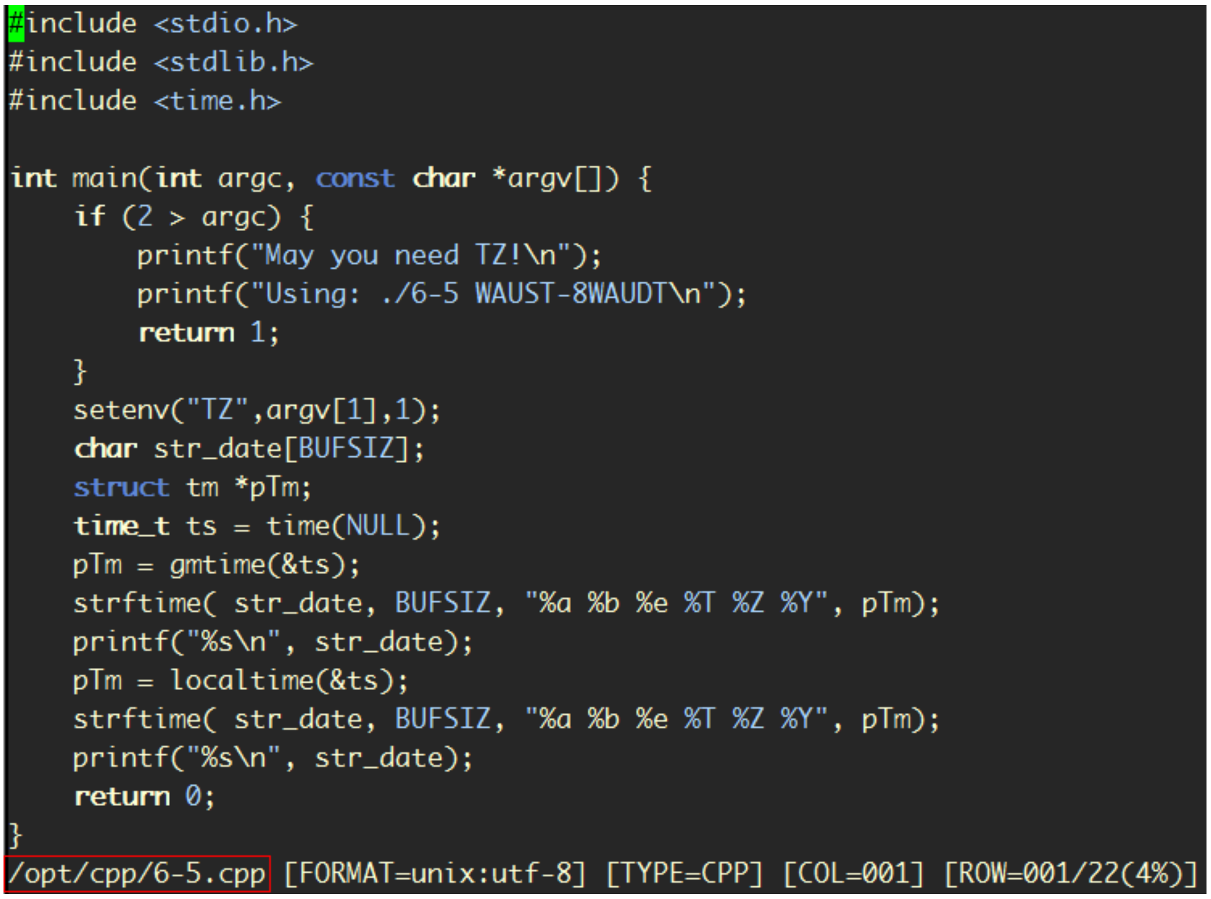
```
mail cpp # echo $(($((16#7FFFFFFFFFFFFFFF))/365/24/60/60+1970))
292471210647      可表示的有效年份（正值）
mail cpp # echo $((16#FFFFFFFF))  可存储的最大年份（包括负值）
4294967295
mail cpp # date -ud "@$((16#7FFFFFFFFFFFFFFF))"  轻松溢出
date: 時間 9223372036854775807 超出可接受的範圍
```



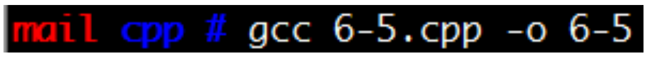
```
struct tm
{
   int tm_sec;
   int tm_min;
   int tm_hour;
   int tm_mday;
   int tm_mon;
   int tm_year;
   int tm_wday;
   int tm_yday;
```

6-5、easy；

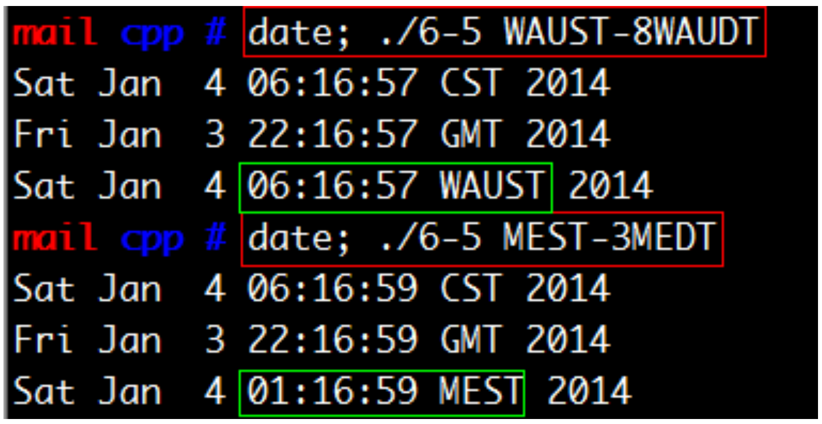
TZ参考地址http://publib.boulder.ibm.com/infocenter/aix/v7r1/index.jsp?topic=%2Fcom.ibm.aix.files%2Fdoc%2Faixfiles%2Fenvironment.htm；



```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main(int argc, const char *argv[]) {
    if (2 > argc) {
        printf("May you need TZ!\n");
        printf("Using: ./6-5 WAUST-8WAUDT\n");
        return 1;
    }
    setenv("TZ",argv[1],1);
    char str_date[BUFSIZ];
    struct tm *pTm;
    time_t ts = time(NULL);
    pTm = gmtime(&ts);
    strftime( str_date, BUFSIZ, "%a %b %e %T %Z %Y", pTm);
    printf("%s\n", str_date);
    pTm = localtime(&ts);
    strftime( str_date, BUFSIZ, "%a %b %e %T %Z %Y", pTm);
    printf("%s\n", str_date);
    return 0;
}
/opt/cpp/6-5.cpp [FORMAT=unix:utf-8] [TYPE=CPP] [COL=001] [ROW=001/22(4%)]
```



```
mail cpp # gcc 6-5.cpp -o 6-5
```



```
mail cpp # date; ./6-5 WAUST-8WAUDT
Sat Jan  4 06:16:57 CST 2014
Fri Jan  3 22:16:57 GMT 2014
Sat Jan  4 06:16:57 WAUST 2014
mail cpp # date; ./6-5 MEST-3MEDT
Sat Jan  4 06:16:59 CST 2014
Fri Jan  3 22:16:59 GMT 2014
Sat Jan  4 01:16:59 MEST 2014
```