strcut stat中各成员含义：

通过man手册查询struct stat的具体构成，附上源代码可知，

struct stat {

dev\_t st\_dev; /\* ID of device containing file \*/

ino\_t st\_ino; /\* inode number \*/

mode\_t st\_mode; /\* protection \*/

nlink\_t st\_nlink; /\* number of hard links \*/

uid\_t st\_uid; /\* user ID of owner \*/

gid\_t st\_gid; /\* group ID of owner \*/

dev\_t st\_rdev; /\* device ID (if special file) \*/

off\_t st\_size; /\* total size, in bytes \*/

blksize\_t st\_blksize; /\* blocksize for file system I/O \*/

blkcnt\_t st\_blocks; /\* number of 512B blocks allocated \*/

time\_t st\_atime; /\* time of last access \*/

time\_t st\_mtime; /\* time of last modification \*/

time\_t st\_ctime; /\* time of last status change \*/

};

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

结构体各个成员表示含义：

st\_dev:文件所在磁盘的磁盘的ID号

st\_ino：节点编号

st\_mode:描述文件的类型和权限

st\_nlink:硬链接的数量

st\_uid:文件属于用户的ID

st\_gid:文件所属用户组ID

st\_rdev:如果是设备文件则是设备ID号

st\_size:文件大小

st\_blksize:文件内容对应的块的大小

st\_blocks:文件内容对应得块数量

st\_atime:上次访问时间，访问文件的操作会改变该值

st\_mtime:上次修改时间，

st\_ctime:上次状态改变的时间，对文件的读写将改变该值。

上述众多的结构体成员中，值得细说的还是属于st\_mode,它不仅包含的文件的类型，也包含的文件的权限信息。通过在某个目录下运行ls命令，出现一个结果，其中某一行开头的十个字符就是文件的类型和属性。

在linux下，一切皆文件，linux将文件分为7个种类，

普通文件:-

目录文件：d

字符设备文件:c

块设备：b

管道文件：p

链接文件：l

套接字文件：s

1

2

3

4

5

6

7

这些信息包含在了st\_mode成员中，通过对其进行解析，可得出一个文件的类型。

具体用法如下：

///判断文件类型，设置相应的标识

if(S\_ISDIR(buf.st\_mode))

type='d';

else if(S\_ISCHR(buf.st\_mode))

type='c';

else if(S\_ISBLK(buf.st\_mode))

type='b';

else if(S\_ISFIFO(buf.st\_mode))

type='p';

else if(S\_ISLNK(buf.st\_mode))

type='l';

else if(S\_ISSOCK(buf.st\_mode))

type='s';

1

2

3

4

5

6

7

8

9

10

11

12

13

14

st\_mode中还包含了文件的权限信息

The following flags are defined for the st\_mode field:

S\_IFMT 0170000 bit mask for the file type bit fields

S\_IFSOCK 0140000 socket

S\_IFLNK 0120000 symbolic link

S\_IFREG 0100000 regular file

S\_IFBLK 0060000 block device

S\_IFDIR 0040000 directory

S\_IFCHR 0020000 character device

S\_IFIFO 0010000 FIFO

S\_ISUID 0004000 set UID bit

S\_ISGID 0002000 set-group-ID bit (see below)

S\_ISVTX 0001000 sticky bit (see below)

S\_IRWXU 00700 mask for file owner permissions

S\_IRUSR 00400 owner has read permission

S\_IWUSR 00200 owner has write permission

S\_IXUSR 00100 owner has execute permission

S\_IRWXG 00070 mask for group permissions

S\_IRGRP 00040 group has read permission

S\_IWGRP 00020 group has write permission

S\_IXGRP 00010 group has execute permission

S\_IRWXO 00007 mask for permissions for others (not in group)

S\_IROTH 00004 others have read permission

S\_IWOTH 00002 others have write permission

S\_IXOTH 00001 others have execute permission

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

上述均为8进制表示的权限，st\_mode成员类型为mode\_t型，深入跟踪，也就是unsigned int 型，一个32位的无符号数，但是，暂时只用到了其中的低16位，其中0-8表示权限，9-11位为属组，12-15位为类型。

若一个文件属性直接用无符号数打印出来出现的是 16877，该数字为十进制，转换为8进制，则表示成040755.

根据文件类型可知040755=0040000+00400+00200+00100+00040+00010+00004+00001=S\_IFSIR+S\_IRUSR+S\_IWUSR+S\_IXUSR+S\_IRGRP+S\_IXGRP+S\_IROTH+S\_IXOTH

即ls -l下开头部分表示 drwxr-xr-x

1

2

3

///获取文件的权限

if(buf.st\_mode & S\_IRUSR)

ownrd='r';

if(buf.st\_mode & S\_IWUSR)

ownwt='w';

if(buf.st\_mode & S\_IXUSR)

ownet='x';

if(buf.st\_mode & S\_IRGRP)

grprd='r';

if(buf.st\_mode & S\_IWGRP)

grpwt='w';

if(buf.st\_mode & S\_IXGRP)

grpet='x';

if(buf.st\_mode & S\_IROTH)

othrd='r';

if(buf.st\_mode & S\_IWOTH)

othwt='w';

if(buf.st\_mode & S\_IXOTH)

othet='x';

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

根据st\_uid和st\_gid或得其对应的用户，

struct passwd \*pa;

pa=getpwuid(buf.st\_uid);

struct group \*gp;

gp=getgrgid(buf.st\_gid);

1

2

3

4

根据文件最后修改时间转换成容易理解的时间表示法

struct tm \*ct;

ct=localtime(&buf.st\_ctime);

1

2

由此，便将struct stat中的大部分成员含义进行了相应的解析，利用该结构体将打开的文件各种信息保存后进行解析，可以实现类似ls -l 的功能。

printf("%c%c%c%c%c%c%c%c%c%c %-d %-s %-s %-ld %-2d %-2d %-d:%-d %s\n",

type,ownrd,ownwt,ownet,grprd,grpwt,

grpet,othrd,othwt,othet,buf.st\_nlink,pa->pw\_name,gp->gr\_name,buf.st\_size,ct->tm\_mon+1,ct->tm\_mday,ct->tm\_hour,ct->tm\_min,filename);

1

2

3

程序运行结果如下图所示，运行结果类似于ls -l 命令的输出。

---------------------

作者：sinat\_16046537

来源：CSDN

原文：https://blog.csdn.net/sinat\_16046537/article/details/51779569

版权声明：本文为博主原创文章，转载请附上博文链接！