Linux操作系统编程

# Linux文件权限管理

# 用户和组ID

- Linux系统通过进程的**有效用户ID**和**有效用户组ID**来决定进程对系统资源的访问权限
- 与一个进程相关联的用户ID和用户组ID有如下几种

| | |
|---|---|
| 实际用户ID<br>实际组ID | 我们实际是谁（即执行程序的用户） |
| 有效用户ID<br>有效组ID<br>附加组ID | 用于文件权限检查 |

**用户ID是进程的属性**

# 用户和组ID

- 通常情况下，有效用户ID等于实际用户ID，有效组ID等于实际组ID；
- 可执行文件的权限中有一个特殊标志，定义为“**当执行此文件时,将进程的有效用户ID设置为文件的所有者**”，与此类似，组ID也有类似的情况。这两个标志位称为：“**设置用户ID**”和“**设置组ID**”，这两位都包含在stat信息中的st_mode中，可用S_ISUID,S_ISGID测试。

# 文件权限

- 对于一个文件的**读权限**决定了我们是否能够打开该文件进行读操作
- 对一个文件的**写权限**决定了我们是否能够打开该文件进行写操作
- 为了在open函数中对一个文件指定O_TRUNC标志，必须对该文件具有**写权限**
- 执行某个可执行文件，都必须对该文件具有**执行权限**

# 目录权限

- 目录文件的**执行权限**也表示可以进入该目录
- 通过文件名打开一个任意类型的文件时，对该文件路径名中包含的每一个目录都应具有执行权限
- 为了在一个目录中创建一个新文件，必须对该目录具有**写权限**和**执行权限**
- 为了删除一个文件，必须对包含该文件的目录具有**写权限**和**执行权限**，对该文件本身则不需要有读、写权限

# 文件权限检查

**●进程访问文件时，内核就进行文件权限检查。这种检查涉及到文件的所有者、文件的所有者所在组、进程有效用户、进程的有效组及进程的附加组。两个所有者是文件的性质，而有效用户与有效组是进程的性质**

**●当进程对某个文件进行操作时，内核按顺序执行下列4步来检查文件权限**

1、若进程的有效用户为root（ID等于0），则允许任何操作；

2、若进程的有效用户等于文件的所有者（ID相同）（即该进程拥有文件）,按照文件所有者具有的权限判定操作是否合法

3、若进程的有效组或进程的附加组之一等于文件所有者所在组,按照文件所有者所在组具有的权限判定操作是否合法

4、按照其他用户具有权限判定操作是否合法

# 文件权限检查

## ●一般情况下

- 若进程有效用户拥有此文件，则按用户权限批准或拒绝该进程对文件的操作；
- 若进程有效用户并不拥有该文件，但进程有效用户属于某个适当的组，则按组权限批准或拒绝该进程对文件的操作
- 若进程有效用户并不拥有该文件，也不属于某个适当的组，则按照其他其他用户权限批准或拒绝该进程对文件的操作

# 新文件和新目录的所有权

- **新文件的所有者设置为进程的有效用户**
- **新文件所有者所在的组，POSIX允许选择下列之一：**
  - 新文件所有者所在的组可以是进程的有效组
  - 新文件所有者所在的组可以是它所在目录的组
- **新文件所有者所在的组取决于它所在目录的设置组ID位是否设置，若设置，则为目录组，否则则为进程有效组**
- **BSD总是用目录组作为新文件所有者所在组**

# access函数

- **函数功能：**按照前述文件权限检查的4个步骤测试存取文件是否具有相应权限
- **函数原型：**int access( const char *pathname, int mode)

| Mode | 说明 |
| --- | --- |
| R_OK | 测试读许可权 |
| W_OK | 测试写许可权 |
| X_OK | 测试执行许可权 |
| F_OK | 测试文件是否存在 |

# chmod、fchmod函数

●**函数原型**

- int  chmod(const char * pathname, mode_t  mode);
- int fchmod( int fd,  mode_t  mode);

●**函数用途：**改变指定文件的权限位。

●**函数说明：**

- chmod要求给出的是文件或目录所在的位置，而fchmod主要针对的是文件，要求调用是相应的文件描述符。
- 修改时，进程的有效用户ID必须等于文件的所有者ID，或是root运行的此进程

# 文件存取许可权

| 常数 | 说明 | 对普通文件的影响 | 对目录的影响 |
| --- | --- | --- | --- |
| S_ISUID | 设置-用户-ID | 执行时设置有效用户ID | （不使用） |
| S_ISGID | 设置-组-ID | 若组执行位设置，则执行时置有效组ID | 将在目录中创建新文件的组ID设置为目录的组ID |
| S_ISVTX | 粘住位 | 在交换区保存程序正文 | 禁止在目录中删除和更名文件 |
| S_IRUSR | 用户读 | 许可用户读文件 | 许可组用户目录项 |
| S_IWUSR | 用户写 | 许可用户写文件 | 许可用户在目录中删除或创建文件 |
| S_IXUSR | 用户执行 | 许可用户执行文件 | 许可用户在目录中搜索给定路径名 |
| S_IRGRP | 组读 | 许可组读文件 | 许可组读目录项 |
| S_IWGRP | 组写 | 许可组写文件 | 许可组写目录项 |
| S_IXGRP | 组执行 | 许可组执行文件 | 许可组在目录中删除或创建文件 |
| S_IROTH | 其他读 | 许可其他读文件 | 许可其他读目录项 |
| S_IWOTH | 其他写 | 许可其他写文件 | 许可其他在目录中删除或创建文件 |
| S_IXOTH | 其他执行 | 许可其他执行文件 | 许可其他在目录中搜索给定路径名 |

# Chmod函数出错信息

| 字段 | 说明 |
| --- | --- |
| EACCES | 给出的文件所处路径没有访问权限 |
| EFAULT | 路径指向的文件地址错误 |
| EIO | 发生I/O错误 |
| ELOOP | 给出的文件所在路径中符号连接过多 |
| ENAMETOOLONG | 路径过长 |
| ENOENT | 文件不存在 |
| ENOMEM | 内核内存空间不足 |
| ENOTDIR | 给出的文件所处路径中包含不是目录的部分 |
| EPERM | 有效用户ID与文件拥有者不同，进程无权访问修改文件权限 |
| EROFS | 文件位于只读文件系统 |

# fchmod函数出错信息

| 字段 | 说明 |
| --- | --- |
| EBADF | 非法的文件描述符 |
| EIO | 发生I/O错误 |
| EPERM | 有效用户ID与文件拥有者不同，进程无权访问修改文件权限 |
| EROFS | 文件位于只读系统 |