操作系统
Operating System

北京理工大学计算机学院
马　锐
Email: mary@bit.edu.cn

版权声明

第9章　Linux文件系统

第9章　Linux文件系统(1)

- Unix文件系统的特点
  - Unix的文件目录采用树形结构
  - Unix将其管理的文件看成一个无结构的字节流
  - Unix把外部设备看成与普通文件一样的特殊文件
- Linux文件系统
  - 类Unix系统
  - 继承了Unix文件系统的特色

4

第9章　Linux文件系统(2)

- Linux最初采用的是Minix的文件系统，但是Minix只是一种试验性的操作系统，其文件系统只支持14个字符的文件名和最大64MB的文件
- 当Linux成熟时，引入了扩展文件系统(Extended File System, Ext FS)，ext文件系统支持255个字符的文件名和最大2GB的文件
- 同时增加了虚拟文件系统(Virtual File System, VFS)，不同类型文件系统通过VFS提供的统一接口进行映射，使得上层应用不依赖于具体文件系统实现

5

第9章　Linux文件系统(3)

- 为了解决性能和ext的一些局限性，1993年引入了ext2，它相当高效和强健，成为广泛使用的Linux文件系统，但存在容错问题
- 2001年，Linux内核2.4.15中引入了ext3，在ext2的基础上引入了日志(Journaling)技术，支持的最大文件为16TB
- Linux内核2.6.19中引入了ext4，与ext3有较大不同，引入了扩展盘区概念，支持的最大文件为1EB

6

9.1 ext2的磁盘涉及的数据结构

- 9.1.1 Linux文件卷的布局
- 9.1.2 超级块
- 9.1.3 块组描述符
- 9.1.4 文件目录与索引节点结构
- 9.1.5 访问控制表ACL

7

9.1.1 Linux文件卷的布局(1)

- 一个ext2磁盘文件卷由若干个大小相等的磁盘块组成，包括1个引导块和n个块组

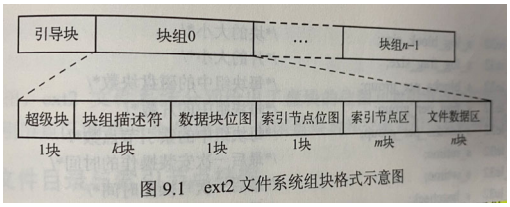


图 9.1 ext2 文件系统组块格式示意图

8

9.1.1 Linux文件卷的布局(2)

- 块（block）
  - 用来存储数据的单元，大小相同
  - block 按照0，1，2，3 的顺序进行编号
  - ext2 支持的 block 的大小有 1024 B、2048 B和 4096 B，其大小在创建文件系统的时候可以通过参数指定
  - block 的大小与数量在格式化完成后不能再改变(除非重新格式化)
  - 每个 block 内最多只能够放置一个文件的数据

9

9.1.1 Linux文件卷的布局(3)

- 引导块(boot block)
  - ext2文件卷的第一个盘块，不使用
  - 用作Linux的引导块或保留不用
  - 用于读入并启动操作系统
- 块组(block group)
  - 每个块组包含相邻磁道的磁盘块
  - 每个块组大小相等且顺序排列
  - 使用“块组描述符”记录块组信息
  - 每个块组中均存放超级块和所有块组信息，但OS只使用块组0中的相应信息

10

9.1.1 Linux文件卷的布局(4)

- 超级块(super block)
  - 存放整个文件卷的资源管理信息
  - 操作系统只使用块组0中的超级块和块组描述符
  - 采用e2fsck程序对文件系统状态进行一致性检查时，将块组0中的超级块和块组描述符复制给其他块组
- 数据块位图(block bitmap)
  - 记录文件数据区各个盘块的使用情况
  - 存放在一个单独的块中

11

9.1.1 Linux文件卷的布局(5)

- 索引节点位图(inode bitmap)
  - 记录索引节点区各个索引节点的使用情况
  - 存放在一个单独的块中
- 索引节点区(inode table)
  - 存放文件的索引节点
  - 一个索引节点存放一个文件的管理和控制信息
- 文件数据区(data block)
  - 存放普通文件和目录文件等

12

9.1.1 Linux文件卷的布局(6)

- 块组个数与大小
  - 一个块组中的盘块和索引节点的位图必须存放在一个单独的磁盘块中
    - 若文件卷大小为8GB，盘块大小为4KB，则每个盘块最多可以记录32K个磁盘块，那么每个块组最大为128MB，文件卷可划分为64个块组
    - 若盘块大小为2KB，则每个盘块最多可以记录16K个磁盘块，那么每个块组最大为32MB，文件卷可划分为256个块组
    - 若文件卷大小为32GB，盘块大小为4KB，则每个盘块最多可以记录32K个磁盘块，那么每个块组最大为128MB，则文件卷可划分为256个块组

13

9.1.2 超级块(1)

- 包含文件的主要管理和控制信息
  - block 与 inode 的总量
  - 未使用与已使用的 inode / block 数量
  - block 与 inode 的大小 (block 为 1, 2, 4K，inode 为 128 bytes)
  - filesystem 的挂载时间、最近一次写入数据的时间、最近一次检验磁盘 (fsck) 的时间
  - 其他一些文件系统的相关信息

14

9.1.2 超级块(2)

```
struct ext2_super_block {
  _le32 s_inodes_count;       //索引节点的总数
  _le32 s_blocks_count;       //盘块的总数
  _le32 s_free_blocks_count; //空闲块计数
  _le32 s_free_inodes_count; //空闲索引节点数
  _le32 s_log_block_size;     //盘块的大小
  _le32 s_blocks_per_group;  //每组中的盘块数
  _le32 s_inodes_per_group; //每组索引节点数
  _le16 s_inode_size; //磁盘上索引节点结构的大小
  ……
};
```

15

9.1.3 块组描述符(1)

- 每个块组都有自己的块组描述符，记录了该块组管理的一些重要信息
  - 盘块位图的块号
  - 索引节点位图的块号
  - 索引节点表的第一个盘块号
  - 块组中空闲盘块数
  - 块组中空闲索引节点个数
  - 组中目录个数
  - ……

16

9.1.3 块组描述符(2)

```
struct ext2_group_desc {
  _le32 bg_block_bitmap;   //盘块位图的块号
  _le32 bg_inode_bitmap;  //索引节点位图的块号
  _le32 bg_inode_table;    //索引节点区
                           //第一个盘块块号
  _le16 bg_free_blocks_count; //组中空闲块个数
  _le16 bg_free_inodes_count; //组中空闲索引节
                              //点个数
  _le16 bg_used_dirs_count;   //组中目录个数
  ……
};
```

17

9.1.4 文件目录与索引节点结构(1)

- 文件目录
  - 采用树形目录结构管理所有文件
  - 把通常的文件目录项分成简单目录项和索引节点两部分
    - 简单目录项包含了文件名和索引节点号等，可以提高文件目录的检索速度
      - 提高了文件目录的检索速度和文件系统的使用效率
    - 文件的管理控制信息存放在索引节点中
      - 系统只保留一个索引节点，就可实现多条路径共享文件，减少信息冗余

18

9.1.4 文件目录与索引节点结构(2)

- 目录项结构

```
struct ext2_dir_entry_2{
  _le32  inode;      //索引节点号
  _le16  rec_len;    //目录项长度
  _u8   name_len;  //实际文件名长度
  _u8  file_type;    //文件类型
  char  name[255]; //文件名是4B整数倍
                     //变长数组
}
```

19

9.1.4 文件目录与索引节点结构(3)

- 文件类型
  - 0，不可知
  - 1，普通文件
  - 2，目录文件
  - 3，字符设备文件
  - 4，块设备文件
  - 5，有名管道
  - 6，套接字文件
  - 7，符号链接

20

9.1.4 文件目录与索引节点结构(4)

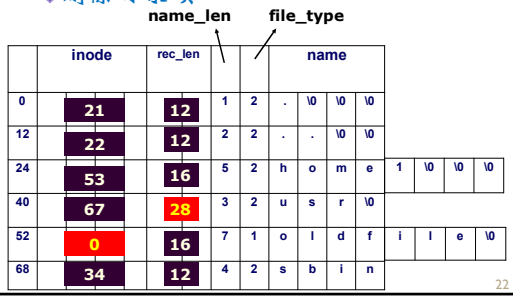
- 文件名
  - name，存放文件名，最多255个字符
  - name_len，文件名实际长度
  - 文件名长度不同，但必须是4字节的整数倍
  - 不足时，以字符'\0‘填充
- 目录项长度
  - 至少占12个字节

21

9.1.4 文件目录与索引节点结构(5)

- 目录文件示例
  - 删除目录项

28=12+16



9.1.4 文件目录与索引节点结构(6)

- 索引节点(inode)
  - 索引节点存放在索引节点区
  - 索引节点区由连续盘块组成
  - 第一个索引块的块号存放在块组描述符的bg_inode_table字段中
  - 一个索引节点的大小是128B
  - 索引节点区占用的块数
    - 一个块组中的索引节点数/每块中的索引节点数
    - 每块中的索引节点数＝块大小/2^7

9.1.4 文件目录与索引节点结构(7)

- 索引节点存放文件的管理和控制信息
  - 该文件的访问模式(rwx)
  - 该文件的所有者与组(owner/group)
  - 该文件的大小
  - 该文件创建或状态改变的时间(ctime)
  - 最近一次的读取时间(atime)
  - 最近修改内容的时间(mtime)
  - 定义文件特性的标志(flag)，如SetUID
  - 该文件真正内容的指向(Pointer)

9.1.4 文件目录与索引节点结构(8)

```
struct ext2_inode {
  _le16 i_mode;          //文件类型和访问权限
  _le16 i_uid;           //拥有者的标识符
  _le32 i_size;          //以字节为单位的文件长度
  _le16 i_gid;           //组标识符
  _le16 i_links_count;   //硬链接计数
  _le32 i_block[15];     //索引表，15×4B
  _le32 i_blocks;        //文件的数据块数
  _le32 i_file_acl;      //文件访问控制表ACL
  ……
};
```

25

9.1.4 文件目录与索引节点结构(9)

- 索引表：是一个有15个元素的数组，每个元素占4B
- 数组的15个元素有4种类型
  - 最初的12个元素是直接索引项，给出文件最初的12个逻辑块号对应的物理块号
  - 索引12是一次间接索引块，对应的文件逻辑块号从12到（b/4）+11，b是盘块大小
  - 索引13是二次间接索引块，对应的文件逻辑块号从b/4+12到 $(b/4)^2+(b/4)+11$
  - 索引14是三次间接索引块，对应的文件逻辑块号从 $(b/4)^2+(b/4)+12$到 $(b/4)^3+(b/4)^2+(b/4)+11$

26

9.1.4 文件目录与索引节点结构(10)

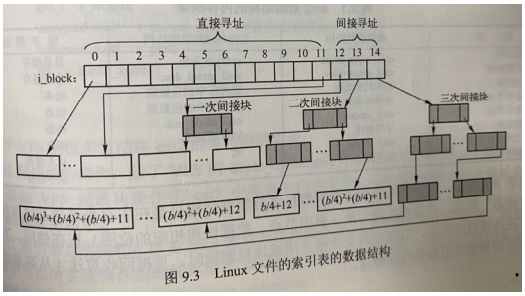


图 9.3 Linux 文件的索引表的数据结构

27

9.1.4 文件目录与索引节点结构(11)

- 符号链接文件
  - 如果文件路径名小于60个字符，直接存放在i_block[]中
  - 如果大于60个字符，需要一个单独的数据块
- 设备文件、管道文件、套接字文件
  - 所有必要信息都存放在索引节点中，不需要额外的数据块

28

9.1.5 访问控制表ACL

```
[root@study doc]# getfacl /root
getfacl: Removing leading '/' from absolute path names
# file: root
# owner: root
# group: root
user::r-x
user:chen:rwx
group::r-x
mask::rwx
other::---
```

29

9.2 ext2的主存数据结构

- 9.2.1 超级块和索引节点对象
- 9.2.2 位图高速缓存

30

9.2 ext2的主存数据结构(1)

- 安装ext2文件系统时，存放在ext2文件卷的磁盘数据结构中的大部分信息被复制到主存RAM中，从而使内核避免了以后的多次重复的磁盘读写操作

表 9.1 ext2 数据结构的内存映像

类 型	磁盘数据结构	主存相应的数据结构	缓 存 模 式
超级块	ext2_super_block	ext2_sb_info	总是缓存
组描述符	ext2_group_desc	ext2_group_desc	总是缓存
块位图	块中的位数组	缓冲区中的位数组	动态
索引节点位图	块中的位数组	缓冲区中的位数组	动态
索引节点	ext2_inode	ext2_inode_info	动态
数据块	字节数组	VFS 缓冲区	动态
空闲索引节点	ext2_inode	无	从不缓存
空闲块	字节数组	无	从不缓存

31

9.2 ext2的主存数据结构(2)

- 频繁使用的超级块和组描述符总是缓存在主存高速缓冲区
- “动态”模式
  - 只要相关的对象（索引节点、数据块和相应的位图）正在使用，其数据就保存在高速缓存中
  - 当相应的文件被关闭或块被删除时，页框回收算法从高速缓存中删除相关数据并将其写回磁盘

32

9.2.1 超级块和索引节点对象(1)

- 当安装ext2文件系统时，用类型ext2_sb_info结构对象存冲ext2_super_block结构，以便内核能找出与该文件系统相关的内容
- 超级块
  - 磁盘结构：ext2_super_block
  - 内存结构：ext2_sb_info

33

9.2.1 超级块和索引节点对象(2)

```
struct  ext2_sb_info {
  unsigned long s_inodes_per_block;  //每块包含索引节点数
  unsigned long s_blocks_per_group;  //每组盘块数
  unsigned long s_inodes_per_group; //每组索引节点数
  unsigned long s_itb_per_group; //每组索引节点区块数
  unsigned long s_gdb_count; //每组的组描述符的盘块数
  unsigned long s_desc_per_block; //每组的组描述符数
  unsigned long s_groups_count; //整个文件系统的组数
  struct ext2_super_block *s_es;    //指向超级块所在
                                    //缓冲区的指针
  unsigned short s_loaded_inode_bitmaps;//索引节点位图
  ……
};
```


9.2.1 超级块和索引节点对象(3)

- 当ext2文件的索引节点对象被初始化时，用类型ext2_inode_info结构对象缓冲磁盘的索引节点ext2_inode结构
- 索引节点
  - 磁盘结构：ext2_inode
  - 内存结构：ext2_inode_info

9.2.1 超级块和索引节点对象(4)

```
struct  ext2_inode_info {
   _u32 i_block[15];    //文件数据块的索引表
   _u32 i_flags;        //文件标志
   _u32 i_file_acl;     //文件访问控制表
   _u32 i_dir_acl;      //目录访问控制表
   _u32 i_block_group;//索引节点所在块组的索引
   _u32 i_next_alloc_block; //预分配文件逻辑块号
   _u32 i_next_alloc_goal; //预分配磁盘物理块号
   _u32 i_prealloc_block;  //预分配第一个数据块
   _u32 i_preallco_count;  //预分配数据块总数
   ……
};
```

9.2.2 位图高速缓存

- ext2中，每一个块组需要两个磁盘块，分别用作描述文件数据区和索引节点区使用情况的位图，即数据区位图块和索引节点位图块
- 随着磁盘容量增加，将所有节点均保存于RAM中不再现实
- 使用大小为EXT2_MAX_GROUP_LOADED的两个高速缓存，分别存放最近访问的大部分索引节点位图和数据块位图

37

9.3 ext2磁盘空间管理

- 9.3.1 磁盘索引节点的管理
- 9.3.2 空闲磁盘块的分配与回收

38

9.3 ext2磁盘空间管理

- 磁盘块和索引节点的分配和回收
- 文件的数据块和其索引节点尽量在同一个块组中
- 文件和它的目录项尽量在同一个块组中
- 父目录和子目录尽量在同一个块组中
- 每个文件的数据块尽量连续存放
- 采用位示图实现磁盘块和磁盘索引节点的分配和回收

39

9.3.1 磁盘索引节点的管理

- 当用户创建一个新文件时，系统为其分配一个磁盘索引节点存放文件的管理控制信息
  - ext2_new_inode()
- 当用户要求从磁盘上删除一个文件或目录时，系统在回收该文件或目录占用的磁盘索引节点之前，先回收文件或目录占用的数据块
  - ext2_free_inode()

40

9.3.2 空闲磁盘块的分配与回收

- 内核为普通文件分配一个磁盘块
  - ext2_get_block()
- 当进程要删除或截短一个文件时，要释放文件不再占用的磁盘块
  - ext2_truncate()

41