Z-World

段武杰

September 8, 2016

内存管理

文件系统

2.1 inode

内核处理文件的关键是 inode，每个文件（和目录）都有且只有一个对应的 inode, 其中包含元数据（如访问权限，上次修改的日期，等等）和指向文件数据的指针。

```
/*
 * Keep mostly read-only and often accessed (especially for
 * the RCU path lookup and 'stat' data) fields at the beginning
 * of the 'struct inode'
 */
struct inode {/* fs.h */
        umode_t                 i_mode;/* 文件访问权限和所有权 */
        unsigned short          i_opflags;
        kuid_t                  i_uid;/* uid about the file */
        kgid_t                  i_gid;/* gid about the file */
        unsigned int            i_flags;

#ifdef CONFIG_FS_POSIX_ACL
        struct posix_acl        *i_acl;
        struct posix_acl        *i_default_acl;
#endif
        /* 负责管理结构性操作（如删除一个文件）和文件相关的元数据例如属性() */
        const struct inode_operations   *i_op;
        struct super_block      *i_sb;
        struct address_space    *i_mapping;

#ifdef CONFIG_SECURITY
        void                    *i_security;
#endif

        /* Stat data, not accessed from path walking */
        /* 对给定的文件系统，唯一的编号标识 */
        unsigned long           i_ino;
        /*
         * Filesystems may only read i_nlink directly.  They shall use the
         * following functions for modification:
         *
         *    (set|clear|inc|drop)_nlink
         *    inode_(inc|dec)_link_count
         */
        union {
                /* 记录使用该 inode 的硬链接总数 */
                const unsigned int i_nlink;
                unsigned int __i_nlink;
        };
        dev_t                   i_rdev;

        loff_t                  i_size;/* 文件大小 */
        struct timespec         i_atime;/* 最后访问时间 */
        struct timespec         i_mtime;/* 最后修改时间*/
        struct timespec         i_ctime;/* inode 最后修改时间 */
        spinlock_t              i_lock; /* i_blocks, i_bytes, maybe i_size */
        unsigned short          i_bytes;
        unsigned int            i_blkbits;
        blkcnt_t                i_blocks;/*指定了按块存放的长度*/

#ifdef __NEED_I_SIZE_ORDERED
        seqcount_t              i_size_seqcount;
#endif

        /* Misc */
        unsigned long           i_state;
        struct mutex            i_mutex;

        unsigned long           dirtied_when;   /* jiffies of first dirtying */
        unsigned long           dirtied_time_when;

        struct hlist_node       i_hash;
        struct list_head        i_io_list;      /* backing dev IO list */
#ifdef CONFIG_CGROUP_WRITEBACK
        struct bdi_writeback    *i_wb;          /* the associated cgroup wb */

        /* foreign inode detection, see wbc_detach_inode() */
        int                     i_wb_frn_winner;
        u16                     i_wb_frn_avg_time;
        u16                     i_wb_frn_history;
```

```
#endif
        struct list_head        i_lru;          /* inode LRU list */
        struct list_head        i_sb_list;
        union {
                struct hlist_head       i_dentry;
                struct rcu_head         i_rcu;
        };
        u64                     i_version;
        atomic_t                i_count;/* 访问该的进程数目inode */
        atomic_t                i_dio_count;
        atomic_t                i_writecount;
#ifdef CONFIG_IMA
        atomic_t                i_readcount; /* struct files open RO */
#endif
        /* 用于操作文件中包含的数据 */
        const struct file_operations    *i_fop; /* former ->i_op->default_file_ops */
        struct file_lock_context        *i_flctx;
        struct address_space    i_data;
        struct list_head        i_devices;
        union {
                struct pipe_inode_info  *i_pipe;
                struct block_device     *i_bdev;
                struct cdev             *i_cdev;
                char                    *i_link;
        };

        __u32                   i_generation;

#ifdef CONFIG_FSNOTIFY
        __u32                   i_fsnotify_mask; /* all events this inode cares about */
        struct hlist_head       i_fsnotify_marks;
#endif

        void                    *i_private; /* fs or device private pointer */
};
```

2.2 inode_operations

大多数请况下，各个函数指针成员的意义可以根据其名称推断。它们与对应的系统调用和用户空间工具在名称方面非常相似。

```
struct inode_operations {
        /* lookup 根据文件系统对象的名称表示为字符串）查找其( inode 实例*/
        struct dentry * (*lookup) (struct inode *,struct dentry *, unsigned int);
        const char * (*follow_link) (struct dentry *, void **);
        int (*permission) (struct inode *, int);
        struct posix_acl * (*get_acl)(struct inode *, int);
        int (*readlink) (struct dentry *, char __user *,int);
        void (*put_link) (struct inode *, void *);

        int (*create) (struct inode *,struct dentry *, umode_t, bool);
        int (*link) (struct dentry *,struct inode *,struct dentry *);
        int (*unlink) (struct inode *,struct dentry *);
        int (*symlink) (struct inode *,struct dentry *,const char *);
        int (*mkdir) (struct inode *,struct dentry *,umode_t);
        int (*rmdir) (struct inode *,struct dentry *);
        int (*mknod) (struct inode *,struct dentry *,umode_t,dev_t);
        int (*rename) (struct inode *, struct dentry *,
                        struct inode *, struct dentry *);
        int (*rename2) (struct inode *, struct dentry *,
                        struct inode *, struct dentry *, unsigned int);
        int (*setattr) (struct dentry *, struct iattr *);
        int (*getattr) (struct vfsmount *mnt, struct dentry *, struct kstat *);
        int (*setxattr) (struct dentry *, const char *,const void *,size_t,int);
        ssize_t (*getxattr) (struct dentry *, const char *, void *, size_t);
        ssize_t (*listxattr) (struct dentry *, char *, size_t);
        int (*removexattr) (struct dentry *, const char *);
        int (*fiemap)(struct inode *, struct fiemap_extent_info *, u64 start,
                      u64 len);
        int (*update_time)(struct inode *, struct timespec *, int);
        int (*atomic_open)(struct inode *, struct dentry *,
                           struct file *, unsigned open_flag,
                           umode_t create_mode, int *opened);
        int (*tmpfile) (struct inode *, struct dentry *, umode_t);
        int (*set_acl)(struct inode *, struct posix_acl *, int);

        /* WARNING: probably going away soon, do not use! */
} ____cacheline_aligned;
```

模板

```c
int main(int argc, char ** argv)
{
        printf("Hello␣world!\n");
        return 0;
}
```