# ext文件系统中的时间戳分析

宝德技术研究院 李磊

在内核的ext文件系统中, inode可以表示一个文件, 其具体结构体如下

struct ext2\_inode {

\_\_le16 i\_mode; /\* File mode \*/

\_\_le16 i\_uid; /\* Low 16 bits of Owner Uid \*/

\_\_le32 i\_size; /\* Size in bytes \*/

\_\_le32 i\_atime; /\* Access time \*/

\_\_le32 i\_ctime; /\* Creation time \*/

\_\_le32 i\_mtime; /\* Modification time \*/

\_\_le32 i\_dtime; /\* Deletion Time \*/

\_\_le16 i\_gid; /\* Low 16 bits of Group Id \*/

\_\_le16 i\_links\_count; /\* Links count \*/

\_\_le32 i\_blocks; /\* Blocks count \*/

\_\_le32 i\_flags; /\* File flags \*/

union {

struct {

\_\_le32 l\_i\_reserved1;

} linux1;

struct {

\_\_le32 h\_i\_translator;

} hurd1;

struct {

\_\_le32 m\_i\_reserved1;

} masix1;

} osd1; /\* OS dependent 1 \*/

\_\_le32 i\_block[EXT2\_N\_BLOCKS];/\* Pointers to blocks \*/

\_\_le32 i\_generation; /\* File version (for NFS) \*/

\_\_le32 i\_file\_acl; /\* File ACL \*/

\_\_le32 i\_dir\_acl; /\* Directory ACL \*/

\_\_le32 i\_faddr; /\* Fragment address \*/

union {

struct {

\_\_u8 l\_i\_frag; /\* Fragment number \*/

\_\_u8 l\_i\_fsize; /\* Fragment size \*/

\_\_u16 i\_pad1;

\_\_le16 l\_i\_uid\_high; /\* these 2 fields \*/

\_\_le16 l\_i\_gid\_high; /\* were reserved2[0] \*/

\_\_u32 l\_i\_reserved2;

} linux2;

struct {

\_\_u8 h\_i\_frag; /\* Fragment number \*/

\_\_u8 h\_i\_fsize; /\* Fragment size \*/

\_\_le16 h\_i\_mode\_high;

\_\_le16 h\_i\_uid\_high;

\_\_le16 h\_i\_gid\_high;

\_\_le32 h\_i\_author;

} hurd2;

struct {

\_\_u8 m\_i\_frag; /\* Fragment number \*/

\_\_u8 m\_i\_fsize; /\* Fragment size \*/

\_\_u16 m\_pad1;

\_\_u32 m\_i\_reserved2[2];

} masix2;

} osd2; /\* OS dependent 2 \*/

};

其中涉及到了4个时间, 分别是atime ,ctime, mtime及dtime, 根据注释这几个时间分别代表

•atime: 最后一次访问此文件的时间

•ctime: 创建文件的时间(注, 内核注释有误, 后面有详细说明)

•mtime: 最后一次修改此文件的时间

•dtime: 此文件被删除的时间

下面我们通过一个实验来验证这几个时间的真正含义,首先通过创建一个空的文件.

dd if=/dev/zero of=ext2.img bs=40M count=1

接着将ext2.img格式化成ext2文件系统

mkfs -t ext2 ext2.img

mkfs –t ext2来创建一个空的ext2文件系统名为ext2.img. 用dumpe2fs查看文件系统的状态

root@ubuntu:~# dumpe2fs ext2.img

dumpe2fs 1.42.9 (4-Feb-2014)

Filesystem volume name: <none>

Last mounted on: <not available>

Filesystem UUID: cffc329d-eecb-44bd-b851-01bd65777b56

Filesystem magic number: 0xEF53

Filesystem revision #: 1 (dynamic)

Filesystem features: ext\_attr resize\_inode dir\_index filetype sparse\_super

Filesystem flags: signed\_directory\_hash

Default mount options: user\_xattr acl

Filesystem state: clean

Errors behavior: Continue

Filesystem OS type: Linux

Inode count: 10240

Block count: 40960

Reserved block count: 2048

Free blocks: 39172

Free inodes: 10229

First block: 1

Block size: 1024

Fragment size: 1024

Reserved GDT blocks: 159

Blocks per group: 8192

Fragments per group: 8192

Inodes per group: 2048

Inode blocks per group: 256

Filesystem created: Tue May 20 23:26:10 2014

Last mount time: n/a

Last write time: Tue May 20 23:26:10 2014

Mount count: 0

Maximum mount count: -1

Last checked: Tue May 20 23:26:10 2014

Check interval: 0 (<none>)

Reserved blocks uid: 0 (user root)

Reserved blocks gid: 0 (group root)

First inode: 11

Inode size: 128

Default directory hash: half\_md4

Directory Hash Seed: a19b5acb-5b13-4636-bc75-864c01195eaf

Group 0: (Blocks 1-8192)

Primary superblock at 1, Group descriptors at 2-2

Reserved GDT blocks at 3-161

Block bitmap at 162 (+161), Inode bitmap at 163 (+162)

Inode table at 164-419 (+163)

7759 free blocks, 2037 free inodes, 2 directories

Free blocks: 434-8192

Free inodes: 12-2048

Group 1: (Blocks 8193-16384)

Backup superblock at 8193, Group descriptors at 8194-8194

Reserved GDT blocks at 8195-8353

Block bitmap at 8354 (+161), Inode bitmap at 8355 (+162)

Inode table at 8356-8611 (+163)

7773 free blocks, 2048 free inodes, 0 directories

Free blocks: 8612-16384

Free inodes: 2049-4096

Group 2: (Blocks 16385-24576)

Block bitmap at 16385 (+0), Inode bitmap at 16386 (+1)

Inode table at 16387-16642 (+2)

7934 free blocks, 2048 free inodes, 0 directories

Free blocks: 16643-24576

Free inodes: 4097-6144

Group 3: (Blocks 24577-32768)

Backup superblock at 24577, Group descriptors at 24578-24578

Reserved GDT blocks at 24579-24737

Block bitmap at 24738 (+161), Inode bitmap at 24739 (+162)

Inode table at 24740-24995 (+163)

7773 free blocks, 2048 free inodes, 0 directories

Free blocks: 24996-32768

Free inodes: 6145-8192

Group 4: (Blocks 32769-40959)

Block bitmap at 32769 (+0), Inode bitmap at 32770 (+1)

Inode table at 32771-33026 (+2)

7933 free blocks, 2048 free inodes, 0 directories

Free blocks: 33027-40959

Free inodes: 8193-10240

接着将ext2.img挂载.

mount -o loop ext2.img temp/

进入temp中, 将cpu信息输出到cpuinfo的文件中.

cat /proc/cpuinfo > cpuinfo

查看cpuinfo的状态

root@ubuntu:~/temp# ll

total 21

drwxr-xr-x 3 root root 1024 May 21 14:33 ./

drwx------ 20 root root 4096 May 21 14:29 ../

-rw-r--r-- 1 root root 3232 May 21 14:33 cpuinfo

drwx------ 2 root root 12288 May 21 14:26 lost+found/

卸载ext2.img

cd .. ; umount temp

接下来我们用winhex来逐步分析.

第一步, 找到文件系统的根目录. 我们知道根目录的inode no是2, 在group 0中, 而group 0 的inode table在164号block上, block size = 1kb, inode size = 128, 这样可以算出根目录的inode的地址

根目录的地址 = 164 \* 1024 + 128 = 0x29080

图1-1, 为根目录inode的16进制

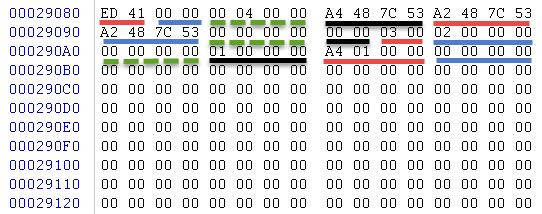


图1-1

第二步, 找到根目录项所在的block, 根据图1-1, 根目录指向的数据块的编号为0x1a4, 故数据块的地址位于0x69000, 如图1-2, 可以看出cpuinfo的inode编号为12.

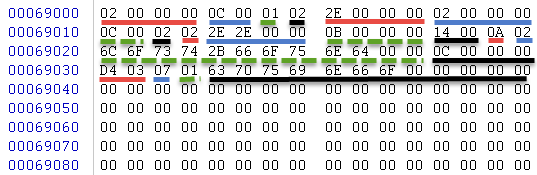


图1-2

第三步, 找到cpuinfo的inode, 其地址为164 \* 1024 + (12 - 1) \* 128 = 0x29580, 如图1-3, 可以看出cpuinfo这个文件的4个时间戳

•atime = 0x537c48a2

•ctime = 0x537c48a2

•mtime = 0x537c48a2

•dtime = 0

atime, ctime, mtime经过换算后的北京时间为‎ 2014‎年‎5‎月‎21‎日‎ ‎14‎:‎33‎:‎06

这和用ls命令看到的时间是一样的.由于cpu这个文件还没被删除过,故dtime为0.

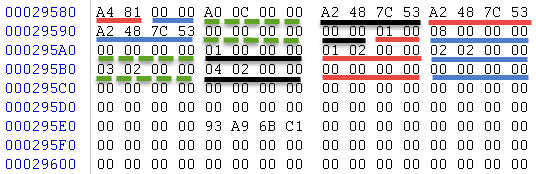


图1-3

接下来,我们分别对atime, ctime, mtime, dtime进行试验

## atime

挂载上ext2.img后, 用cat命令来对cpuinfo这个文件进行访问

cat cpuinfo

卸载ext2.img, 分析cpuinfo的inode

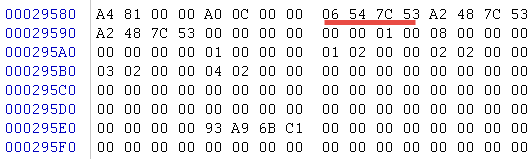


图1-4

可以看出用cat 访问后, atime变成了0x537c5406, 即北京时间‎2014‎年‎5‎月‎21‎日‎ ‎15‎:‎21‎:‎42.

## ctime

挂载上ext2.img后, 将cpuinfo的权限改为777

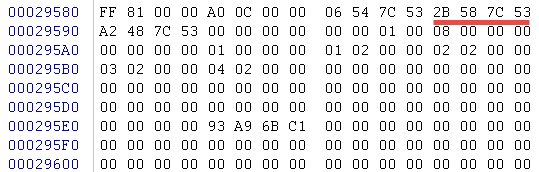


图1-5

这时, ctime竟然改变了,变成北京时间‎2014‎年‎5‎月‎21‎日‎ ‎15‎:‎39‎:‎23了,故这里的ctime不是creation time, 而是change time.

## mtime

挂载上ext2.img后, 使用echo对cpuinfo进行修改

echo hello >> cpuinfo

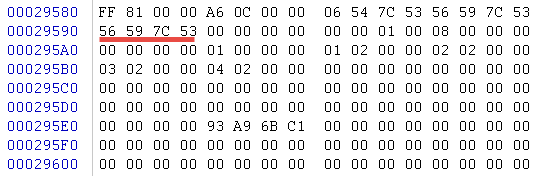


图1-6

mtime变成了北京时间‎ 2014‎年‎5‎月‎21‎日‎ ‎15‎:‎44‎:‎22

## dtime

挂载上ext2.img后,删除cpuinfo, 这时候可以发现cpuinfo的inode中的所有数据块的指向全部为0, dtime由0变为0x537c5a5b, 即北京时间 ‎2014‎年‎5‎月‎21‎日‎ ‎15‎:‎48‎:‎43.

rm –rf cpuinfo

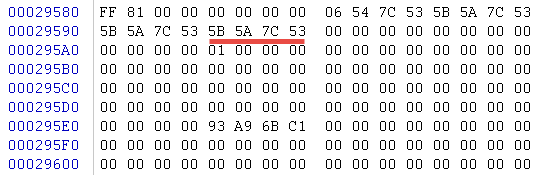


图1-7

## Windows与linux间的时间差异

在windows中有三中时间记录, 创建时间, 修改时间, 访问时间. 其中创建时间在linux的ext文件系统中不存在, 修改时间和linux的ext文件系统一致. 当在linux ext文件系统中修改一个文件的权限或者读出文件内容, 在windows中的文件浏览器中都被视为修改了访问时间.