CS202: File System II

Scheduling Note

11/21 Tuesday ✓
11/23 Thursday ✗ (Turkey, Pie, etc.)
11/28 Tuesday ✗
11/30 Thursday ✓

Last Class

file → inode
("a") (#11)

Directories

inode 11

metadata

- owner
- perm
- links
- timestamp

Direct

indirect

Data Block
Lab 2

open<in \rightarrow read<in \rightarrow struct dirent

\ldots

```
struct dirent 2

ino_t
char
ushort
\ldots
```

_inode:\leftarrow Inode Number
d-name:\leftarrow file name

Directory

```
Metadata
Slot (3)

mode = D...
owner
perm
nlink
timestamp
\ldots

indirect
```

Really a dirent

From the kernel's perspective

- File Is An Inode
- Names Are In Directories

File System Layout (abstract)

- Intermediate Step To Explore Directory Layout
Possible directory organizations

@ Flat: Single Directory For All

/ - a
   b
   c
   ...

Problems?

@ Directory Per User
BENEFITS?

PROBLEMS

Hierarchical

/home

/home/apanda

/home/lineeman

/home/...
```
/home/apanda> rm b ; rm a
rm: 
/home/apanda> ln -s a c
```

**FAST FILE SYSTEM**

Learning: Anatomy of a more complex file system

Goal: Used by BSD until recently

- OpenBSD uses an extended version (FFS 2)

```
stat("/a/b") -> 42
```
- Directories
  [Files in a directory often accessed together; think I/O]
- Files
  [Good for sequential access]

How to allocate blocks better.

Cylinders: Our friend from two classes ago.

Observations:
- More storage (sectors) than on a single track. Why?
- No seek necessary when reading from the same cylinder. Why?

Cylinder Group

Core abstraction: combine blocks from 1 or more adjacent cylinders

<table>
<thead>
<tr>
<th>CG</th>
<th>S</th>
<th>Desc.</th>
<th>Block Bitmap</th>
<th>Inode Bitmap</th>
<th>Inodes</th>
<th>Data Blocks</th>
</tr>
</thead>
</table>
Bitmaps: A quick way to track what blocks & inodes are free

You will implement ONE IN
STEP 1 OF LAB 5

0 | 1 | 2 | 3 | 4
00000010
00000001

0 | 1 | 2 | 3
00000000

0 | 1 | 2

0 Allocating directory
  \rightarrow Place in cylinder group with largest # of free inodes

0 Allocating file inode
  \rightarrow If possible, place in same cyl. group as parent directory
  (Hash to choose random cyl. group otherwise)
Data blocks

File < 48 kB: In same cyl. group as inode

> 48 kB: Redirect to another cyl. group; prefer one with large number of free data blocks

Buffer Cache

Why multiple super blocks?
**VFS**

- `/boot`
- `/usr`
- `/home`
- `/mnt`
  - `/mnt/usb`
  - `/mnt/cdrom`
  - `/mnt/dropbox`

**mount**

```python
file = open("/home/out.txt")
open("/mnt/dropbox/out.txt")
```

- **inode** → File metadata, authoritative copy on disk.
  - Layout & Contents Dictated By File System

- **vnode** → Logical representation in memory, providing common interface for the OS.
  - Layout & Contents Dictated By Kernel
struct fuse_operations fs_oper = {
    .getattr = fs_getattr,
    .readlink = fs_readlink,
    .mknod = fs_mknod,
    .mkdir = fs_mkdir,
    .opendir = fs_open, // No difference between open and opendir.
    .readdir = fs_readdir,
    .unlink = fs_unlink,
    .rmdir = fs_rmdir,
    .symlink = fs_symlink,
    .rename = fs_rename,
    .link = fs_link,
    .chmod = fs_chmod,
    .chown = fs_chown,
    .truncate = fs_truncate,
    .open = fs_open,
    .read = fs_read,
    .write = fs_write,
    .statfs = fs_statfs,
    .fsync = fs_fsync,
    .ftruncate = fs_ftruncate,
    .fgetattr = fs_fgetattr,
    .utimens = fs_utimens,
};