VFS and A Simple File System

CS333 S20 :: Meeting 05

Course Logistics

- Lab 1
  - Due tonight. Anyone Stuck?
  - TA session tonight with Jacob
- Next class: FUSE
  - Documentation is scattered across many locations, as is tradition with open-source projects…
  - We will use FUSE in a “lab session” next Thursday with our guest Tom

Course Logistics

- Lab 2: **Hello FUSE**
  - Build “restricted functionality” file system using FUSE:
    - Allow reading and writing to 1 file only
    - Can’t create new files or delete existing files
  - We will be using “Panic” machines in TCL 312
    - Given “root” privileges; use non-CS accounts
    - (required) partners for lab due to limited machines && because it is likely how you will be working in the “real world”

Last Class

- HDDs
  - Geometry
  - Caching
  - Scheduling

This Class

- VFS and Simple File system
  - Similar in practice to some ***real*** file systems (FFS, ext4)
  - This is the design we will implement later in Lab

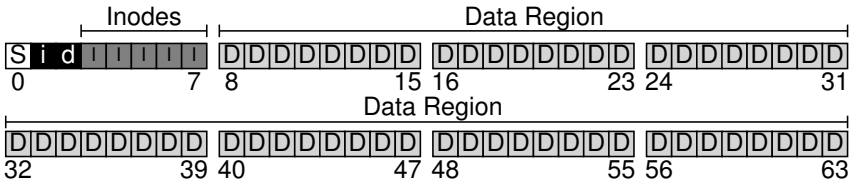
Key VFS Data structures

- `inode`
  - Persistent information about a single file
  - “Index” node (indirection node?)
- `superblock`
  - Persistent information about entire file system
- Allocation structures (several types)
  - Free list, bitmap, extent list, etc.

Simple File System: On-Disk Layout

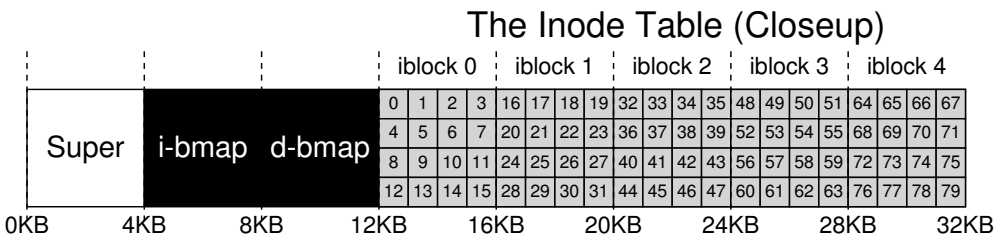
- Space partitioned into data and metadata
  - Superblock is located in the first block
    - Why?
  - Static inode table
  - Static data block allocation bitmap

Simple File System: On-Disk Layout



(OSTEP Chapter 40)

Simple File System On-Disk Layout



(OSTEP Chapter 40)

Example inode Fields (ext2)

Size	Name	What is this inode field for?
2	mode	can this file be read/written/executed?
2	uid	who owns this file?
4	size	how many bytes are in this file?
4	time	what time was this file last accessed?
4	ctime	what time was this file created?
4	mtime	what time was this file last modified?
4	dtime	what time was this inode deleted?
2	gid	which group does this file belong to?
2	links_count	how many hard links are there to this file?
4	blocks	how many blocks have been allocated to this file?
4	flags	how should ext2 use this inode?
4	osd1	an OS-dependent field
60	block	a set of disk pointers (15 total)
4	generation	file version (used by NFS)
4	file_acl	a new permissions model beyond mode bits
4	dir_acl	called access control lists

Figure 40.1: **Simplified Ext2 Inode**

(OSTEP Chapter 40)

Access Patterns: open("/foo/bar"), read()

	data bitmap	inode bitmap	root inode	foo inode	bar inode	root data	foo data	bar data[0]	bar data[1]	bar data[2]
open(bar)			**read**			**read**				
				read						
							read			
					read					
read()					**read**					
					write			**read**		
read()					**read**					
					write				**read**	
read()					**read**					
					write					**read**

Access Patterns: Creating a File

	data bitmap	inode bitmap	root inode	foo inode	bar inode	root data	foo data	bar data[0]	bar data[1]	bar data[2]
create (/foo/bar)			**read**			**read**				
				read						
		read					**read**			
		write								
							write			
					read					
					write					
				write						
write()	**read**				**read**					
	write							**write**		
					write					
write()	**read**				**read**					
	write								**write**	
					write					
write()	**read**				**read**					
	write									**write**
					write					

Figure 40.4: **File Creation Timeline (Time Increasing Downward)**

VFS Handout
(sync up at end)

Course Logistics: Up Next

- Lab 2 (Hello, FUSE!)
  - Fill out partner survey (linked on course “labs” page)
  - Tom will help to answer questions during lab
- **I’ll be gone next Monday afternoon until Sunday (FAST conference)**