Project 4 Part 2 - Adding a new file system type to Minix

- Brandon Tarney
- May 3 2017
- JHU EP CS 605.412
- Project 4 Part 2 - Adding a new file system type to Minix

How to create a new filesystem type

New file systems are created with the `mkfs` command meaning we will need a new implementation of this command. The file system is identical to v3 Minix filesystem, aside from the file system type which is embedded in the superblock of the file system when it is created. First we would need to update the makefile for the appropriate program (i.e. PROG= mkfs.mfshw4). Next we modify the magic # (essentially the file system type) contained in the super block which can be found in sbin.mkfs/v3/const.h (#define SUPER_V3 0x4d5a) as well as the macro SUPER_MAGIC which points to SUPER_V# (we simply point to our renamed SUPER_V3 macro). This "magic number" is stored in Super.H struct super_block superblock.s_magic and is used to pre-calculate s_inodes_per_block.

How to create a disk with that new filesystem

`MKFS` is the standard unix utility which creates a file system on a fresh disk. It's typically located on /sbin/. The exact use of this utility would look something like `mkfs.hw4 -t <type> <partition>` . We will use the file system type type we just created and any partition available.

How to create a new filesystem service associated with the new filesystem

I should probably have mentioned by now we should rename all the #ifndef & @define statements which use "MFS" and replaced them with "fs_h4" or something like it. There are many constants and variables we should probably rename. We might reconsider some macro's as well, such as mfsdir.h MFS_DIRSIZ which controls the directory size (name). All of that being said, the more serious changes are modifying the makefile first and foremost to build our service (fs_hw4, instead of mfs). Once again we must modify const.h to include a macro defintion of our file system type and its associated magic number. We can also modify Super.c to add the new file system type and show our file system only supports our new fs type. Finally, we would also want to update many print statements, such as error states which print "mfs" statements, ours would use "hw4_fs" or something like that (for example, see read_super in super.cm, or misc.c::fs_new_driver).

How to modify the 'mount' command to mount the new filesystem within the Minix file system hierarchy

The mount command is the final step to acutallying using hw4_fs. First we will need to mount the file system to an existing directory (`mkdir <dir>`). Then we can use the command `mount <file system_special_file> <directory_mount_point>`. The mount command itself (/usr/src/minix/commands/mount/mount.c) would need to be modified in several ways. We would want to change the file system macro by defining the HW4_FS_TYPE to be "hw4" or something like that. Next we would want to add a FSVERSION_H4 to the switch case which attempts to auto-detect the FS type and/or version. For our case we would assign type to be "HW4_FS_TYPE". Ultimately we may need to update /usr/src/minix/lib/libc/gen/fsversion.c since it is responsible for checking the FS version/type (responsible for retrieving the magic number). We would simply add FSVERSION_HW4. As with past modifications, there are many printf statements in /usr/src/minix/fs/mfs/mount.c which would need to be updated to reflect our new FS type.