Universidad Tecnológica Centroamericana

Sistemas Operativos I

Period 2 2015

Using SmallFS FUSE Linux Driver

Ivan de Jesús Deras

10 de Mayo del 2015

1 Introduction

SmallFS FUSE driver can be used to read, write, delete and rename files in a small file system used in our Operating System class. So in order to use the driver you have to install the FUSE development package available for you distribution. Then you have to download the source code from GitHub `https://github.com/ideras/fuse-smallfs`.

2 File System Description

The main purpose of a file system is to keep a record of the names and sectors of files on the disk. The file system in this operating system is managed by two sectors at the beginning of the disk. The Disk Map sits at sector 1, and the Directory sits at sector 2. This is the reason your kernel starts at sector 3.

The Map tells which sectors are available and which sectors are currently used by files. This makes it easy to find a free sector when writing a file. Each sector on the disk is represented by one byte in the Map. A byte entry of 0xFF means that the sector is used. A byte entry of 0x00 means that the sector is free. You will not need to read or modify the Map in this project since you are only reading files in this project; you will in the next.

The Directory lists the names and locations of the files. There are 16 file entries in the Directory and each entry contains 32 bytes (32 times 16 = 512, which is the storage capacity of a sector). The first six bytes of each directory entry is the file name. The remaining 26 bytes are sector numbers, which tell where the file is on the disk. If the first byte of the entry is 0x0, then there is no file at that entry.

For example, a file entry of:

```
4B 45 52 4E 45 4C 03 04 05 06 00 00 00 00 00 00 00 00 00 00...
K  E  R  N  E  L
```

means that there is a valid file, with name KERNEL, located at sectors 3, 4, 5, 6. (00 is not a valid sector number but a filler since every entry must be 32 bytes).
If a file name is less than 6 bytes, the remainder of the 6 bytes should be padded out with 00s.

You should note, by the way, that this file system is very restrictive. Since one byte represents a sector, there can be no more than 256 sectors used on the disk (128kB of storage). Additionally, since a file can have no more than 26 sectors, file sizes are limited to 13kB. For this project, this is adequate storage, but for a modern operating system, this would be grossly inadequate.

3 Compiling SmallFS

To compile SmallFS follow these steps:

1. Go to the folder where you downloaded the SmallFS source code using the command:
   `cd <SmallFS Source Folder>`
2. Type the command `make`
3. If the compilation process suceed you'll have two additional folder `bin/` and `obj/`. SmallFS executable file is in the `bin/` folder and it's called `sfs`.

4 Using SmallFS

Now you can use SmallFS driver to access the files in the floppy image, take into account that in order to use the floppy image, this should have the initial map and root directory. To do this you'll need two additional files: `map.img` and `dir.img`. These contain a Map and Directory for a file system consisting of only the kernel. To create the initial file system follow these steps:

```
dd if=map.img of=floppya.img bs=512 count=1 seek=1 conv=notrunc
dd if=dir.img of=floppya.img bs=512 count=1 seek=2 conv=notrunc
```

This sets up your initial file system.

Then you can follow these steps to mount the floppy image:

1. Create a new folder called `sfs_root/` using the command `mkdir sfs_root`
2. Mount the file system using the command
   <SFS source folder>`/bin/sfs` *<path to floppy image>* `sfs_root/ -f -s`
3. Now you can navigate to the folder `sfs_root` using a file manager or the console to read, write, delete and rename files in the floppy image.
4. When you're done with the changes to the file system, you have to unmount it using the command:
   `fusermount -u sfs_root`