EEE335 – Lab

FAT File System

Introduction

The file system controls how data is backed up, organized, and retrieved on a disk. The file system allows you to abstract low-level details by implementing the file and folder concepts. There are several different file systems such as FAT (Windows), NTFS (Windows), Ext (Linux), UFS (Unix), and Mac OS Extended (Mac) that you may be familiar with. The purpose of this lab is to familiarize you with the FAT file system. This system was developed by Microsoft in 1977, but several versions have been developed over the years: FAT12 (1980), FAT16 (1984), FAT32 (1996) and ExtFAT (2006). The version or variant number refers to the number of table element bits which in turn represent the addressable size of the clusters (a contiguous memory space). The increased number of table element bits corresponds to an increase in storage space.

Nowadays, the FAT file system is often used by default on flash cards and USB flash drives. When you format your key, Windows will choose the most appropriate FAT version depending on the size of the key. In this lab, you will analyze a USB key formatted in FAT16.

Aim

The aim of this lab is to familiarize you with the FAT16 file system. You will learn how to:

1. identify the organization of the FAT16 file system;
2. decode directory entries;
3. browse the directories;
4. use the FAT structure to identify the clusters used by a file; and
5. recover a deleted file.

Description of the FAT16 File System

The organization of the FAT16 file system is depicted below in Figure 1.

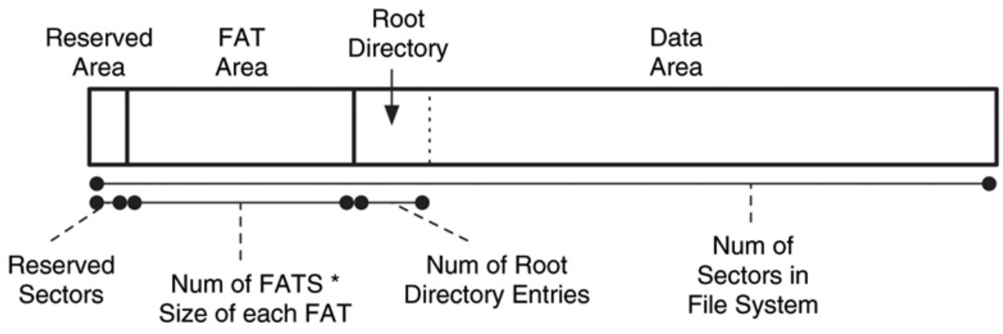
Reserved Area
FAT Area
Root Directory
Data Area
Reserved Sectors
Num of FATS * Size of each FAT
Num of Root Directory Entries
Num of Sectors in File System

Figure 1 – FAT16 File System Structure

The layout of the FAT file system has three sections:

1) Reserved area – boot sector;
2) FAT area – File Allocation Table structures; and
3) Data area.

Each area is made up of a number of sectors, usually 512 bytes in size and this value will be specified in the boot sector. The boot sector is always in sector 0 of the partition. The number of reserved sectors (which includes the boot sector) is specified in the boot sector, in FAT16 there is normally only one reserved sector.

After the reserved sector(s), there are FAT (File Allocation Table) structures. There are usually two FATs, a primary and a backup and the number of FATs will be defined in the boot sector. These two FATs are always identical and provide redundancy in case the disk is damaged.

In the FAT16 system, the root folder immediately follows the FATs at the beginning of the data area. This folder contains 32-byte directory entries, with the number of entries in the root folder specified in the boot sector. This number is used to calculate the size of the root folder.

The units used to store data are referred to as clusters. A cluster is a group of consecutive sectors and the number of sectors that form a cluster must always be a power of 2. The cluster size will be defined in the boot sector. Each cluster is given an address and the first cluster is always 2; clusters 0 and 1 do not exist (that's just the way it is). Also, only the data area uses cluster addresses. Since the position of a file or folder is given by the number of its first cluster, it is necessary to use the following equations to calculate its position in sectors or bytes.

$$Sector = [(Cluster - 2) * NumSectorsPerCluster + Sector\ of\ Cluster\ 2] \quad (1)$$

$$Bytes = Sector * 512 \quad (2)$$

The structure of the boot sector is given in Figure 2 below. Note that when multiple bytes are used to encode a number, the encoding is done in Little-Endian format, which means that the least significant byte is in the smaller address. You must therefore reverse the order of the bytes before reading the value.

[0-2]	Jump to boot code
[3-10]	OEM name in ASCII
[11-12]	Bytes per sector
[13-13]	Sectors per cluster (powers of 2 and smaller than 32KB)
[14-15]	Size in sectors of the reserved area
[16-16]	Number of FAT structure, typically 2
[17-18]	Maximum number of entries in root directory
[19-20]	16-bit value of number of sectors in file system
[21-21]	Media type (0xf8 for fixed disks and 0xf0 for removable)
[22-23]	16-bit size in sectors of each FAT
[24-25]	Sectors per track
[26-27]	Number of heads
[28-31]	Number of sectors before start of partition
[32-35]	32-bit value of number of sectors in file system (either this value or the 16-bit value must be 0)

Figure 2 - Structure of FAT16 Boot Structure

In FAT file system, a directory is like a file; it has a name and a first cluster. The contents of the directory is a list of entries. Each directory entry describes a child file or folder. The inputs have a size of 32 bytes and their structure is given in Figure 3.

[0-0]	First char of file name in ASCII (0xE5 or 0x00 if unallocated)
[1-10]	Characters 2 to 11 of file name in ASCII
[11-11]	File attributes (read only, hidden, system file, volume label, long file name (LFN), Directory, Archive)
[12-12]	Reserved
[13-13]	Created time (tenths of seconds)
[14-15]	Created time (hours, minutes, seconds)
[16-17]	Created day
[18-19]	Accessed day
[20-21]	Always 0 for FAT12/16
[22-23]	Written time (hours, minutes, seconds)
[24-25]	Written day
[26-27]	First cluster address
[28-31]	Size of file (always 0 for directories)

Figure 3 - Structure of a FAT16 Directory Entry

As we saw in class, the directory entry identifies only the first cluster used by the child file or directory. It then use the File Allocation Table (FAT) to find the other clusters by following the chain of pointers. The size of the entries in the FAT table is identified by the file system version. The FAT16 system therefore has 16-bit inputs. The relationship between a directory entry, the clusters used, and the FAT table is shown in Figure 4 below.

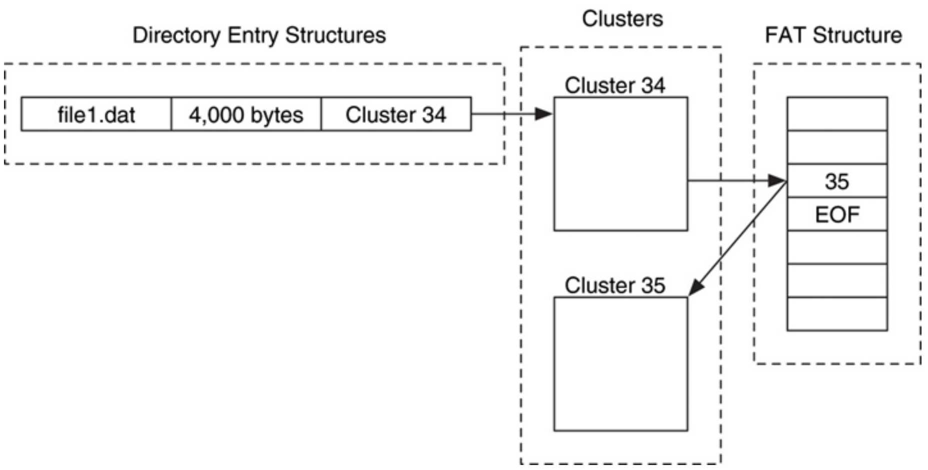

Figure 4 - Relationship between Directory Entry Structure, Clusters and FAT Structure

Preparation of your Lab Environment

You will complete this lab on the EEE435_xubuntu virtual machine that you have been using since the beginning of the course. Before you start the VM, you must connect it to the Internet. In VirtualBox, right-click on EEE435_xubuntu and select Settings. Select the Network tab, then Adapter 1. In the menu, select NAT. You can now start your VM. Open a web browser and confirm that you are connected to the internet.

In this lab, you will analyze the contents of a USB key formatted in FAT16. However, instead of giving each of you a physical key, the key has been imaged for you using the `dd` application. You can download the image here.

Place it on the desktop (or a folder of your choosing) within your VM. To connect this image to your system, use the following commands:

```
$ unzip FAT_16_64mb.zip
$ sudo kpartx -arv FAT16_64mb.dd
$ sudo mkdir ~/FAT16
$ sudo lsblk | grep 64M
```

Note the number of the loop device associated with the 64 MB disk, you will need to use this number in the following command (the number you need is below in red text, and note that the number on your OS may be different).

```
$ sudo mount /dev/loop5 ~/FAT16
```

A MYUSB icon should automatically appear on your desktop. You can navigate the folders to become familiar with the content, but make sure you do not edit anything.

To analyze the image, you will use the Active Disk Editor software which is a specialized hexadecimal editor for disk analysis.

Start Active Disk Editor by clicking `Start Menu -> Active Disk Editor`. Enter the password `password` to start the application with root permissions. In the Getting Started window, select `Open Disk Image`. Navigate to select the image * .dd file. You may need to change the file type to `All Files` (bottom right of the *Open Disk Image File* window) in order for * .dd files to appear. Once opened, you will see the raw data of the file system that was saved to disk.

If the Template window does not open automatically, on the main navigation bar, select `View->Windows->Templates` so that you can use the applicable FAT file system templates.

Disk Analysis

Using Active Disk Editor to analyze the image and answer the following questions:

Question 1: Analyze the boot sector and identify:

- Size of sectors in bytes: ____________________
- Size of clusters in bytes: ____________________
- Size of partition in sector: ____________________
- Number of reserved sectors: ____________________
- Number of FATs: ____________________
- Number of sectors per FAT: ____________________
- Number of entries in the root directory: ____________________
- Size of the root directory in sectors: ____________________

Question 2: Identify the sectors used in each of the following:

- Boot sector: ____________________
- Reserved area: ____________________
- FAT 1: ____________________
- FAT 2: ____________________
- Root directory: ____________________
- Data section: ____________________

Question 3: Where is the root "\" directory located?

- Sector: ____________________
- Byte: ____________________

Question 4: Where is the "\SONGS" directory located?

- Cluster: ____________________
- Sector: ____________________
- Byte: ____________________

Question 5: Analyze the directory entry for the file "\SONGS\MYFAVS>TXT" and using the FAT table, identify:

- The size of the file in bytes: ____________________
- The clusters used by this file: ____________________
- Its byte location: ____________________

In the FAT file system, when a file or directory is deleted, the directory entry is not cleared, the first letter of the name is simply replaced by 0xE5. In addition, the contents of the file in the clusters are not erased, only the entries in the FAT table that point to these clusters are reset to 0 to identify that the clusters are free. A deleted file that is not fragmented on the disk can therefore easily be recovered.

Question 6: Return to the "\SONGS" folder, you will find two deleted files. First analyze the file "\SONGS\åICHELLE.TXT". Provide the following:

- The size of the file in bytes: ____________________
- The cluster used by the file: ____________________
- Its byte location: ____________________

Question 7: Inspect the contents of the file and guess what its original name was:

____

Question 8: Now analyze the second file "\SONGS\åETBACK.TXT" and identify:

- The size of the file in bytes: ____________________
- The cluster used by the file: ____________________
- Its byte location: ____________________

Question 9: Since the file is deleted, the FAT table should identify that the cluster used by the file is free with the corresponding value 0x0000. Visit the FAT table and identify:

- The addresses of these two bytes in the FAT table: ____________________
- The content of these two bytes: ____________________
- What does this signify: ____________________
- What is the file content: ____________________
- Can this file be retrieved: ____________________

Challenge

Now that you are more familiar with the FAT16 file system, here is a challenge. You will find in the folder `\MYSTUFF` a deleted file. A secret word has been saved at the end of this file. Your task is to use *Active Disk Editor* to modify the data on the disk in order to recover the file. To activate the changes, click the `Edit` button and select `Allow Edit Content`. When done, click the `Save` button. Minimize *Active Disk Editor* and go to your desktop. Right-click on `MYUSB` and select `Unmount`.

Enter `sudo mount /dev/loop5 ~/FAT16` to remount the image. Open `MYUSB` now and navigate to the `\MYSTUFF` folder. You should see the recovered file. Open it to find the secret word.

Notify an instructor when you have recovered the file.

Question 10: What changes did you have to make to the file system to recover the deleted file. Be specific and include screenshots that may be useful in your explanation.

Lab Submission

This Lab is optional, but presented for those wishing to pursue knowledge! (Thanks Captain Corona!)