

CSIS0230A Principles of Operating Systems(Class A)

Tutorial 12

Examining an ext2 filesystem

In this tutorial, we will use a utility called `lde` (Linux Disk Editor) to examine the contents of a disk partition, and tries to recover a deleted file in it. To be on the safe side, we will not use a real partition, but instead use a regular file for that purpose.

1. Preparation

Login as `root`, and create and populate a filesystem on the file `test.img` as follows:

1. Create a file `test.img` of size 40MiB, containing all zeros. This can be done by using the `dd` command, copying from the device file `/dev/zero` with a block size of `1M` and a count of `40`. See the man page of `dd` for details, in particular the `if`, `of`, `bs` and `count` options.
2. Use `mke2fs test.img` to create a filesystem there.
3. Create a directory `testmnt`, and mount the newly created filesystem to it using `mount -o loop test.img testmnt`. Type `mount` to make sure that the filesystem is mounted.
4. Change into the `testmnt` directory, and type `tar xzvf ~/t12test.tgz` to extract `t12test.tgz` to your filesystem. Type `sync` to make sure all data are written out.
5. Delete the file `lde/changelog` in the `testmnt` directory.
6. `cd` out of the `testmnt` directory and `umount` it.

2. Starting and using lde

`Lde` is designed to recover deleted files, but it works well for investigating the filesystem as well. You can download it from its web site <http://lde.sourceforge.net>, but to speed things up, it has already been installed in computers of our lab. To start `lde`, type `lde` followed by the name of the device or file that holds the filesystem (i.e., `test.img`).

After a beginning message (which you should bypass by pressing a key), the program greets you with the **superblock view**, which display information contained within the superblock. Now **use the filesystem size to derive the information you see**. Note that a “zone” means a data block.

Other than the superblock view, `lde` has three other primary views: inode view, block view, and recovery view. You can switch between them by typing `s`, `i`, `b` and `r` respectively, and can exit the program from a primary view by typing `q` (be careful **not** to use it unless you want to quit!). **The four primary views are independent** in the location that is displayed. The program maintains a current inode number and a current block number, which are shown in the title bar.

The block view: in the block view, you can see a **hex-dump of a portion of the disk**. This way you can read arbitrary information within the filesystem. Within this view, you can move the cursor around by using the cursor keys, page-up and page-down, and when the cursor is move across a block boundary, the current block number is updated. If you want to jump to a particular block, you can type `#`, which allow you to type in an arbitrary block number (add `$` before the number if you want to input in hexadecimal).

The inode view: in the inode view, you can see all the information stored within the current inode. Remember that an inode stores all information about a file, directory, device, pipe, socket or symbolic link, except the filename which is stored as data of the directory. Arrow keys will

bring you around the fields of the view. If you want to view another inode, you can use page-up and page-down, or you can type # and enter an inode number.

The recovery view: this is used for recovering files. It stores a “fake inode”, to contain direct block and indirect block numbers. In the recovery view, you can dump all these blocks to a file by typing `r`. In the inode or block view, if you find a block that you want to save, you can type a key corresponding to the characters displayed in the top-right corner of the title bar. This copies the block number to the fake inode. If you are in the inode view, you can copy a whole inode to the fake inode by typing `R`.

The directory popup: in the inode and block view, if the current block is a directory, you can show a directory popup by typing `d`. In the directory popup, you can find information stored within a directory block in an easy-to-read format. You can navigate the directories by using up and down arrows to select a directory and typing `enter` to switch to it, or you can type `q` to get back to the previous primary view.

Shortcuts: in the block view, inode view and directory popup, if the cursor is currently at a number, you can assign it to the current inode number and jump to the inode view by the `I` key, and you can assign it to the current block number and jump to the block view by the `B` key.

3. Your tasks

After familiarizing yourselves with `lde`, do the followings:

1. Show the group descriptors of group 0 (i.e., first group) in the block view. By using the tutorial notes, read out the group descriptor information.
2. Determine which block is used for the block bitmap of group 0, and show it in the block view. Determine which data blocks are used.
3. Repeat step (2) for the inode bitmap of group 0.
4. Find the inode table of group 0 in the block view. Where is the end of that inode table? (Hint: you need to know the number of inodes in each group, which can be found in the superblock, i.e., block 1.)
5. Go to the inode view and visit inode 2, the reserved inode for the root directory. (Inode 1–10 are reserved, many for things that are not yet implemented. Look at `/usr/include/linux/ext2_fs.h` if you want to know what they are reserved for.) Compare the block view of the data block and the directory popup.
6. Find the inode for the file `lde-2.6/UNERASE`. What blocks are used for its content? Using the block view, show the content of the first few blocks and the indirect block.
7. Find the inode for the symbolic link `/lde`. How is the link target stored? (Hint: look at the block numbers in the inode.)
8. There is a hard link in the `lde-2.6` directory. Can you find it? How it differs from the symbolic link?
9. Find the inode for the deleted `changelog` file. Try to recover it into a new file of the `/root` directory.