CS3500: Operating Systems

Lab 12

Date: 8th Nov 2019

The goal of this lab is to be understand the file system design within xv6, and then to support a new addressing scheme from inodes to data blocks. In particular, support will be added for extent based files.

This lab is for 10 marks with 2 additional marks available as bonus. Submit a single patch file for the code along with a PDF report summarising your work.

1. (2 points) Study the inode structure in xv6 to identify the largest file size supported. Create a userspace program `create_file.c` that creates a file of a given size. Create a file of the maximum size possible. What happens when you append to the file to increase beyond this maximum size?

2. (5 points) Currently xv6 supports three kinds of files: regular files, directories, and devices. Add a fourth category called `DUP_BLK_FILE` which has the following behavior: Whenever a block with address `BNO` is referred to by the `addrs` array of the file's inode, then it follows that both blocks `BNO` and `BNO+1` are reserved for that file. Thus with a single entry in `addrs` two blocks are reserved for the inode. You do not need to change the format of the blocks as stored within the indirect block.

   Supporting this new file type requires changes whenever the array `addrs` is accessed, primarily in the file `fs.c`. Note that you are not to change the existing code, but only add additional code to be executed conditionally if the file category is `DUP_BLK_FILE`. Now modify the `open, read, write` system calls to take an additional parameter to create, read, and write a file of type `DUP_BLK_FILE`.

   With this modification, compute the largest file possible and create a file of that size using `create_file.c`.

3. (5 points) Create a new file type called `EXT_FILE` which has the following behavior: Each pair of values stored in the inode's `addrs` array store the starting and ending block numbers of a contiguous set of blocks which contain the data. Thus, two entries in the `addrs` array can store any interval of blocks to be referenced by the inode. Again, the format of the indirect block need not be changed.

   Like in the previous question, modify the `open, read, write` system calls to work with the file type `EXT_FILE`.

   With this modification, create a file of size 100 MB.