CS342

Operating Systems

Project 4

Muhammad Arham Khan
21701848
Date: 27th May, 2020

PROGRAM 1

- ## Creating file:

I used N = 100000 to make the file and the file size was 410.0 MB and the file name produced is “output”.

- ## Formatting file:

I used the command mkfs.ext4 output 4096 to format the file. The output was:

mke2fs 1.44.1 (24-Mar-2018)
Discarding device blocks: done
Creating filesystem with 110000 4k blocks and 110080 inodes
Filesystem UUID: 94ff068c-440c-4bff-9189-99e057fd60fe
Superblock backups stored on blocks:
32768, 98304

Allocating group tables: done
Writing inode tables: done
Creating journal (4096 blocks): done

As shown in the output, 110080 inodes were created in this part.

- ## Mounting the file:

To mount the file, I created a directory called “mount” and then used the command “sudo mount output mount” to mount the output file generated earlier to the created directory.

In the mounted directory, I created several files and folders to test. And upon mounting, they were visible as follows:

```
root@mak:/home/mak/Desktop/CS342Project4/mount# ls
lost+found  test2.txt  test3.txt  test4  test5  test.txt
```

But after running the “umount mount” command, the output was empty as the virtual disk was not mounted anymore.

- ## Information about file:

Upon running the command, “dumpe2fs output”, the following output was observed:

dumpe2fs 1.44.1 (24-Mar-2018)
Filesystem volume name: <none>

Last mounted on: /home/mak/Desktop/CS342Project4/mount
Filesystem UUID: 3e52f478-ccda-4fc5-b64a-ff582aa98e8a
Filesystem magic number: 0xEF53
Filesystem revision #: 1 (dynamic)
Filesystem features: has_journal ext_attr resize_inode dir_index filetype extent 64bit flex_bg sparse_super large_file huge_file dir_nlink extra_isize metadata_csum
Filesystem flags: signed_directory_hash
Default mount options: user_xattr acl
Filesystem state: clean
Errors behavior: Continue
Filesystem OS type: Linux
Inode count: 1024
Block count: 4096
Reserved block count: 204
Free blocks: 2892
Free inodes: 1008
First block: 1
Block size: 1024
Fragment size: 1024
Group descriptor size: 64
Reserved GDT blocks: 31
Blocks per group: 8192
Fragments per group: 8192
Inodes per group: 1024
Inode blocks per group: 128
Flex block group size: 16
Filesystem created: Wed May 27 19:54:06 2020
Last mount time: Wed May 27 20:05:08 2020
Last write time: Wed May 27 20:07:30 2020
Mount count: 4
Maximum mount count: -1
Last checked: Wed May 27 19:54:06 2020
Check interval: 0 (<none>)
Lifetime writes: 127 kB
Reserved blocks uid: 0 (user root)
Reserved blocks gid: 0 (group root)
First inode: 11
Inode size: 128
Journal inode: 8
Default directory hash: half_md4
Directory Hash Seed: 22654a9b-b40c-48a4-8a37-eea87e94d1bd
Journal backup: inode blocks
Checksum type: crc32c
Checksum: 0xc4999424
Journal features: journal_64bit journal_checksum_v3
Journal size: 1024k
Journal length: 1024

Journal sequence: 0x0000000d
Journal start: 0
Journal checksum type: crc32c
Journal checksum: 0xe0c8da9f

Group 0: (Blocks 1-4095) csum 0x9279 [ITABLE_ZEROED]
Primary superblock at 1, Group descriptors at 2-2
Reserved GDT blocks at 3-33
Block bitmap at 34 (+33), csum 0xdfe0a978
Inode bitmap at 50 (+49), csum 0x1d0150f2
Inode table at 66-193 (+65)
2892 free blocks, 1008 free inodes, 4 directories, 1008 unused inodes
Free blocks: 1204-4095
Free inodes: 17-1024

This output shows that, there are:
2892 free blocks, one group (group 0), the bitmap is in the 34th block, Bitmap is big enough to hold information about all blocks in group, Inode Bitmap is on the 50th block and occupies 16 blocks, Inode Table of "Group 0" is on the 66th block and occupies 127 blocks.

PROGRAM 2

The sample directory used for the output is '/home/mak/Desktop/CS342Project3' and the output generated is:

root@mak:/home/mak/Desktop/CS342Project4# ./p2 /home/mak/Desktop/CS342Project3
File or Dir Name: .vscode
File size: 0
File type: 0
Inode number: 0
number of blocks: 0
User ID: 0
.....
File or Dir Name: module1.o
File size: 0
File type: 0
Inode number: 0
number of blocks: 0
User ID: 0
.....
File or Dir Name: ..

File size: 4096
File type: 16877
Inode number: 398003
number of blocks: 8
User ID: 1000
.....
File or Dir Name: makefile
File size: 138
File type: 33204
Inode number: 393264
number of blocks: 8
User ID: 1000
.....
File or Dir Name: module3.o
File size: 138
File type: 33204
Inode number: 393264
number of blocks: 8
User ID: 1000
.....
File or Dir Name: module2.c
File size: 138
File type: 33204
Inode number: 393264
number of blocks: 8
User ID: 1000
.....
File or Dir Name: 21701848
File size: 138
File type: 33204
Inode number: 393264
number of blocks: 8
User ID: 1000
.....
File or Dir Name: inputfile
File size: 138
File type: 33204
Inode number: 393264
number of blocks: 8
User ID: 1000
.....
File or Dir Name: module1.c
File size: 138
File type: 33204
Inode number: 393264
number of blocks: 8
User ID: 1000

.....
File or Dir Name: 21701848.tar.gz
File size: 138
File type: 33204
Inode number: 393264
number of blocks: 8
User ID: 1000
.....
File or Dir Name: app
File size: 138
File type: 33204
Inode number: 393264
number of blocks: 8
User ID: 1000
.....
File or Dir Name: module2.o
File size: 138
File type: 33204
Inode number: 393264
number of blocks: 8
User ID: 1000
.....
File or Dir Name: module3.c
File size: 138
File type: 33204
Inode number: 393264
number of blocks: 8
User ID: 1000
.....
File or Dir Name: .
File size: 4096
File type: 16893
Inode number: 393244
number of blocks: 8
User ID: 1000
.....

PROGRAM 3

In this part, I used a file called “sampleFile” generated from p1 (the size of this file was 41.0MB). To access the file, I used the command “./p3 5 sampleFile” so basically, 5 was K and sampleFile was the filename. The file was randomly accessed 100 times and the following results were obtained:

1st Run:

Average random access time: 1.59 microsecond

Running after restarting OS:

Average random access time: 1.42 microsecond

Running after cleaning dump:

Average random access time: 11.16 microseconds

Learnings:

It is evident that there wasn't much change between the first run and when the system was restarted. But when the dump was cleared out, the time increased dramatically to (11.16 ms). This behaviour can be justified by removal of the file cache on cache dump which resulted in retrieval of file from physical memory again and it took extra time. On reboot, this probably did not happen as the file was loaded to cache again and was readily available for access.