Operating System
Jacobs University Bremen
Dipak Kandel

Homework 6

Problem 6.1

a) The new directory contained in the new file system is Lost&Found. Files that would normally be lost because of directory corruption(or a file has a inode but is not linked via any paths) would be linked in the new filesystem's lost+found directory by inode number.

b) After a new filesystem has been created a certain portion(blocks) is reserved for the root user. So although the no of free block is 881(in my case) Available is 677. It had not been the case than a regular user may use up all the free space and we can no more work with the filesystem as it is already out of space. Thus, with some space reserved for root user, one can clean or do a filesystem repair.

c) No change is seen in the file system. But the disk partition is vulnerable to random writing as it is no more occupied as the file is already removed.

d) No of free blocks remains same and no of free inodes is decremented by 1(as creating a file creates a inode).
Although the size of big.data is 4194304(seek offset(4096) * block size(1024) but no of free blocks does not change. Here, dd seek asks the system to create an offset of 4096 blocks. while performing so on UNIX system a sparse file is created. The advantage of sparse files is that storage is only allocated when actually needed: disk space is saved. Here, we are just creating an offset means creating a 'hole' of 4096 block sizes, which information is stored as a metadata of file thus the size of file seems to be 4194304 but actually none of the disk is written, hence the free blocks remains same.

e) The chattr +i sets the i attribute. Files with i attributes are immutable. That is the file can not be modified. Hence rm is not successfull.

f) chroot command changes temporarily the root directory of file system.
It is important to copy a statically linked version as we are in a new file system and have changed the root directory(hence don't have access to other files which maybe stored in the normal file system /bin directory) to the mounted folder. if it was dynamically linked version then program can not access to required library files or required files, which may be residing anywhere in the original filesystem. Hence everything that the program needs must be static.

Problem 6.2

a) The over/top is stored in /upper directory.
If we append over/low, the modification are seen in over/low and upper/low(which is physically same location). The modification does not effects lower/low.// Once we unlink over/low the file would be removed from over/low and upper/low but no change is seen in lower/low. The upper directory would simulate this removal by creating a "whiteout" file which exits only in overlay filesystem directory.
changing permission of over/lo changes permission of instance of lo file at over/lo and upper/lo. The original instance of file remains unmodified.

b) live CD's creation, Creating docker and kubernetes like system for container management and orchestration.

c) No, the overlay file system does not always copy data when metadata of a file in the lower layer is changed. It only does so when metadata only copy up feature is enabled. Yes, it is possible to stack multiple layers, and can be acheived by using the the colon (":") as a separator character between the directory names;
