EEL--5737 Principles of Computer System Design

Homework #2

Nirali Patel

Part A

A1. One way to speed up the resolving of names is to implement a cache that remembers recently looked-up (name, object) pairs.

- **a.** What problems do synonyms pose for cache designers, as compared with caches that don't support synonyms?
- **b.** Propose a way of solving the problems if every object has a unique ID.

a. The problem that synonyms pose is that in cache, it might be the case that there are multiple names pointing same object. Sometimes, it does the similar problem while mapping various virtual address to a physical address in cache memory. The space being utilized is more and unnecessary. The cache space is getting used and is prone to any manipulations. The change in any object with name gets carried to other name as well.

b. The object can be assigned unique ID. This indices or extra information with the object will help user to know name, object and a unique ID. Then, if someone wants to modify object by name, it can access for that ID and modify it or else invalidate the copies.

A2. Louis Reasoner has become concerned about the efficiency of the search rule implementation in the Eunuchs system (an emasculated version of the unix system). He proposes to add a referenced object table (ROT), which the system projects will maintain for each session of each user, set to be empty when the user logs in. Whenever the system resolves a name through of use a search path, it makes an entry in the ROT consisting of the name and the path name of that object. The "already referenced" search rule simply searches the ROT to determine if the name in question appears there. If it finds a match, then the resolver will use the associated path name from the ROT. Louis proposes to always use the "already referenced" rule first, followed by the traditional search path mechanism. He claims that the user will detect no difference, except for faster name resolution. Is Louis right?

No. There will be other differences as well apart from faster name resolution. The associated path name from the ROT might refer to some other file.

For instance, there is a file abc.txt located at following location '/home/fusepy/abc.txt'. The reference or the path will be saved in ROT for the corresponding name. Now, if some user searches for a file with same name but located in some different location, say '/home/downloads/abc.txt'. And, because the name is same it will follow the "already referenced" rule. Therefore, the method by Louis will prevent user to find the correct file located in downloads and will instead give the file from earlier address.

A3. i) Suppose blocks are 4Kbytes (4096 Bytes) in size, and there are 4 inodes per block. Assume an inode uses 8 Bytes to store type and reference count; the rest of the inode space is used to store block numbers. Suppose, the disk has 2^32 blocks, and 1024 blocks are used to store inodes.

a. What is the largest number of files that can be stored?

The disk file system uses 1024 blocks to store inodes. And one block has four inodes. So, the total inodes for files are 1024*4=4096. The maximum files could be stored are 4096. But, we have one root inode that will already store address of root directory. Therefore, 4096-1=4095 are the largest files that can be stored.

b. What is the largest file size that can be stored (in Kbytes)?

Each block size is of 4Kbytes. 4096 bytes are divided for 4 inodes per block, which comes to 1024 bytes for one inode. The 1024 bytes also have 8 Bytes of metadata, i.e to store reference count and store type. 1024-8 Bytes=1016 Bytes of space is available for inode to store block address. A address takes 32 bits i.e, 4 bytes. So, address takes 4bytes and 1016/4=254 is the maximum number of blocks that can be addressed using one inode. Then, 254* 4KB=1016KBytes will become the largest size that can be stored.

ii)

a. What is returned on LOOKUP("file2.txt",2)?

The inode number 2 here, refers to the block number 20. Since, block number 20 is not available for searching file2.txt. It can either return failure if the file2.txt is not available or if available it returns the corresponding integer inode number.

b. What is returned on LOOKUP("file1.txt",5)?

3 is returned on looking for file1.txt. The '16' block number is referred from the inode number 5. When we look for file1.txt, we can find 3 as the inode number.

c. What is returned on LOOKUP ("Hello world!",5)?

The block number referred with '5' as inode number is 16. Since, the block number 16 doesn't have "Hello World!". It would return failure.

iii) Suppose you create a new hard link "xyz" in directory"/data" that links to "/data/file3.txt". Describe (in words, or use a drawing based on the figure above) the new state of all blocks that may have changed due to this operation.

The hard link 'xyz' links to '/data/file3.txt'. This will increase reference count in 15th inode by **one**.
5th inode looks like:

File1.txt	3
File3.txt	15

Back	7
Onemore	3
Xyz	**15**

iv) Assume the sequence of events for two processes (A and B) running in the same computer where this file system is mounted:

Time T1: A issues fdA=open("/data/file3.txt"), and obtains fdA=3

Time T2: B issues fdB=open (" /data/file3.txt") and obtains fdB=3

Time T3: thread A issues read(fdA,bufA,n=5)

Time T4: thread A issues close (3)

Time T5: thread B issues read(fdB,bufB,n=5)

What values(if any) are returned in the buffer bufA of thread A at time T3, and bufB in thread B at time T5?

The file descriptors for the two processes (A and B) is 3. According to question, bufA will have n=5 characters in it. So, the bufA at Time T3 will have **"Hello"**.

And, bufB will have the next five characters from the cursor placed at the last thread process **" Worl".**

v) Suppose the system has a second hard disk, with a second file system as depicted in the figure below. Suppose you mount this second file system under directory "/mnt" of the root file system. Is it possible to create a hard link "file4.txt" under "/data" that links to "file4.txt" in the second file system? Is it possible to create a symbolic link "file4.txt" under "/data" that links to "file4.txt" in the second file system? Describe (in words, or use a drawing based on the figure above) the new state of all blocks that may have changed due to these operations, if they are possible.

A hard link cannot be created that links 'file4.txt' under '/data' in one file system to another file system. As hard link binds inode number to the name. Whereas, it was made possible by soft link.

A symbolic link can be created of 'file4.txt' under '/data' that will link to 'file4.txt' of second file system. The symbolic link created would be shown in block number 16 with any inode number assigned by the first file system. The meta data information will be located in the file system 1 having al details of inode assigned to it.

References

https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&cad=rja&uact=8&ved=0ahUKEwjZ3avapbTWAhXCZiYKHf_TAVYQFggtMAE&url=http%3A%2F%2Fweb.mit.edu%2F6.033%2F1995%2Fhandouts%2Fpostscript%2Fh24.ps&usg=AFQjCNHg0VBGK6pdSwLnV5nV8VEQi__Tng

https://www.ece.cmu.edu/~ece447/s13/lib/exe/fetch.php?media=onur-447-spring13-lecture24-advancedcaching-afterlecture.pdf

http://www.cse.unsw.edu.au/~cs9242/02/lectures/03-cache/node8.html#SECTION00082000000000000000

Principles of Computer System Design (By-Jerome. H. Saltzer and M. Frans Kaashock)