JAYNESH PATEL

300433093

CYBR371

TASK 1:

wellingtonCl	administrator	doctor	nurses	receptionist	other
patients	rwx	r-x	r-x	r-x	r-x
Dt08031993	rwx	r-x	r-x	r-x	
Le07041970	rwx	r-x	r-x	r-x	
Mi23122001	rwx	r-x	r-x	r-x	
Pa13091976	rwx	r-x	r-x	r-x	
pbasicinfo.log	rwx	r-x	r-x	r-x	
pmedicalrecord.log	rwx	r-x	r-x	r-x	
scripts	rwx	---	---	---	---
acls.sh	rwx	---	---	---	
audit.sh	rwx	---	---	---	
file-system.sh	rwx	---	---	---	
patients.sh	rwx	---	---	r-x	
searchpatient.sh	rwx	r-x	r-x	r-x	
staff-and-acl.sh	rwx	---	---	---	
visit.sh	rwx	r-x	---	---	
staff	rwx	r-x	r-x	r-x	r-x
administrator	rwx	r-x	r-x	r-x	
doctors	rwx	r-x	r-x	r-x	
nurses	rwx	r-x	r-x	r-x	
receptionist	rwx	r-x	r-x	r-x	
sbasicinfo.log	rwx	r-x	r-x	r-x	

Note: each of the patients have their own individual pbasicinfo.log and pmedicalrecord.log but i didn't want to list it for each of the patients listed as they would all be the same.
This is also the same for the staff directory as well. Each staff member object has their own sbasicinfo.log and the permissions are the same.

4. List the permissions and ACLs for all subjects/objects in the system. Explain the followings:

- Do these ACLs match the access matrix? List any deviations from the access matrix
- Explain any design decisions you had to make and the reason behind it

wellingtonCl	administrator	doctor	nurses	receptionist	other
patients	rwx	r-x	r-x	rwx	r-x
Dt08031993	rwx	r-x	r-x	rwx	
Le07041970	rwx	r-x	r-x	rwx	
Mi23122001	rwx	r-x	r-x	rwx	
Pa13091976	rwx	r-x	r-x	rwx	
pbasicinfo.log	rwx	r-x	r-x	rwx	
pmedicalrecord.log	rwx	r-x	r-x	rwx	
pmedicalrecord.log		rwx (Assigned Doctor)			
scripts	rwx	---	---	---	---
acls.sh	rwx	---	---	---	
audit.sh	rwx	---	---	---	
file-system.sh	rwx	---	---	---	
patients.sh	rwx	---	---	r-x	
searchpatient.sh	rwx	r-x	r-x	r-x	
staff-and-acl.sh	rwx	---	---	---	
visit.sh	rwx	r-x	---	---	
staff	rwx	r-x	r-x	r-x	r-x
administrator	rwx	r-x	r-x	r-x	
doctors	rwx	r-x	r-x	r-x	
nurses	rwx	r-x	r-x	r-x	
receptionist	rwx	r-x	r-x	r-x	
sbasicinfo.log	rwx	r-x	r-x	r-x	

I have made slight changes to the access matrix above which wasn't there at the start when I designed it.

- The doctors ACL's over pmedicalrecord.log.
- The receptionist ACL's over the Patients directory. Receptionist owns the patients directory.
- Otherwise all of the ACL's listed insided acls.sh are the same as the access matrix above.

The change from the access matrix is that the assigned doctors have their own permissions to their own patients that they are treating. This is because the assigned doctors need to be able to write to the .log file as they are the ones that are treating/prescribing the patients so they need to write the additional information. This means that multiple doctors can have RWX permissions to one patient's pmedicalrecord.log if the patients registered doctor isn't available to treat them and so they are assigned a doctor. They are assigned a doctor through the receptionist. (You cannot execute a .log file from my knowledge, so giving just RW- permissions would've done the same)

We have also changed the owner of the patients directory to the receptionist as this will allow the receptionist to have full access to anyone that is registered as a patient and have rwx to the patients.sh. This is because having the receptionist being able to access everything within the patients directory but also not being able to see certain information about the patient's pmedicalrecord.log was difficult to implement when assigning doctors permissions when running scripts as receptionist.

5. Explain in detail where the ACL information of an object is saved on a Linux system and how your system keeps track of them.

Disk/system is divided into equally sized blocks that can store data, each of the blocks can be assigned to any file. If the file is larger than the block assigned then the information is divided into multiple blocks. Each file has an associated inode to it, that stores which blocks make up the content of that file. All inodes are the same size and are stored in an array in the superblock which contains all inodes in the system. Because all inode sizes are consistent they make it easy to recall and look up.

Each inode has blocks and metadata. If you want the data of a file, the inode will tell you all the blocks that contain that data. If the file is too big for one inode block it will store more information within another assigned inode block and reference it with a pointer these are called indirect blocks. However if you want to know more about the file itself it will recall and tell you the metadata and this is where the ACL's are stored.

The linux filesystem is made up of the structure of inodes. Inodes contain metadata that relate to the files within the file system. Each inode in the file system represents an individual file or multiple if they have the same content and data. The inode then contains the metadata about the file such as file size, device ID, UID, GID, ownership/mode, timestamps, reference counts (how many hard links point to this) and pointer to the actual data. However it doesn’t store the name of the file, this is stored separately along with the inode number. So it can link into the metadata that represents the file. This allows the system to have multiple file names that points to the one piece of data, this is also called a hardlink.

ACL’s are bits of information that are directly associated with a file system object. ACL’s stands for Access Control List. They allow the specific permissions rights for the file systems, and provide a more flexible way of managing the access rights for each user. ACL’s allow you to give permissions for any of users or groups to anything within the disk. Basically they allow you to set a list of access permission for a file or directory.

The location of the ACL is saved within the metadata of inode that represents that file. Within these inodes they have a field called i_shadow. If an inode has a stored ACL the i_shadow field will point to a shadow inode. Within the file system shadow inodes are used like regular files. Each shadow inode stores an ACL in its data block (metadata). The same principle applies as stated above where multiple files with the same ACL permissions could point to the same shadow inode which has their ACL permissions.

‘I’ in inode stands for index

https://www.usenix.org/legacy/events/usenix03/tech/freenix03/full_papers/gruenbacher/gruenbacher.pdf