CYBR371 – Assignment One
greenthom – 300536064

Q.1.

	setup.sh	grades.xlsx	final/questions.pdf	final/solutions.pdf	final/student000/answers.docx	final/student001/answers.docx
arman	.	rw	rw	rw	r	r
ilona	.	rw	r	r	r	r
student000	.	...	r	..	rw	...
student001	.	...	r	..	...	rw
student002	.	...	r	..	...	...

Q.2. setup-cybr371.sh provided
Q.3.

In a filesystem, the ownership of directories is important for security and access control. Assigning ownership to the root user provides security and system integrity which is why each directory has root as user owner with read, write, and execute permissions. Assigning ownership of directories to root helps maintain the integrity of the system by preventing unauthorized changes that could alternate the main operation of the filesystem.

```
root@ubuntu:/opt# getfacl cybr371
# file: cybr371
# owner: root
# group: lecturer
user::rwx
group::rwx
other::r-x
```

The base cybr371 directory is user-owned by root with read write and execute access. It is also group owned by lecturer with the same permissions which allows the lecturer group to add content within the cybr371 directory. Read and execute access is granted to other which means that groups tutor and students can access and read the contents of that directory. This is like all sub-directories within (e.g. assignment1, final, lab3) in cybr37. My thought for this is deleting a file (or empty directory) requires write access to the parent directory, which you must be the owner of. Deleting a non-empty directory requires that you empty the directory first, which the lecturer, tutor, and student groups should not be able to do. Given in this scenario, setting the owner to root is acceptable, due to the permissions granted align with the principle of least privilege and meet the requirements of the system. Since

```
root@ubuntu:/opt/cybr371# ls -al
total 0
drwxrwxr-x  11 root lecturer  240 Apr 14 14:59 .
drwxr-xr-x   1 root root       80 Apr 14 14:59 ..
drwxrwxr-x  96 root lecturer 1960 Apr 14 14:59 assignment1
drwxrwxr-x  96 root lecturer 1960 Apr 14 14:59 assignment2
drwxrwxr-x  96 root lecturer 1960 Apr 14 14:59 final
-rwxrw----+  1 root lecturer    0 Apr 14 14:59 grades.xlsx
drwxrwxr-x  96 root lecturer 1960 Apr 14 14:59 lab1
drwxrwxr-x  96 root lecturer 1960 Apr 14 14:59 lab2
drwxrwxr-x  96 root lecturer 1960 Apr 14 14:59 lab3
drwxrwxr-x  96 root lecturer 1960 Apr 14 14:59 lab4
drwxrwxr-x  96 root lecturer 1960 Apr 14 14:59 lab5
drwxrwxr-x  96 root lecturer 1960 Apr 14 14:59 midterm
```

lectures, tutors and students still have read and execute access, they can interact with the directories and access the necessary course materials while maintaining the integrity of the directories. Granting lectures to add content within each directory with write permissions allows lecturer to have a say in the course content without having overall access control over the entire filesystem. Inside the cybr371 directory tutors and students cannot create files or directories within it which maintains the layout of the course structure but can however read what is within the directories to be aware of its contents.

```
root@ubuntu:/opt/cybr371# getfacl grades.xlsx
# file: grades.xlsx
# owner: root
# group: lecturer
user::rwx
group::rw-
group:tutor:rw-
mask::rw-
other::---
```

The grades.xlsx file is owned, and group owned by root and lecturer respectively. Tutors also have read-and-write access to this file so they can mark students and provide them with their grades. This helps ensure that lectures and tutors have the necessary access to efficiently manage and update student grades throughout the course. Others have been set to --- so students are not able to access, write, or delete the grade file which keeps the integrity and security of the grading system as allowing students to modify the grades would lead to unauthorized changes, tampering, or manipulation of their or other grades. Giving lectures and tutors only read-write access ensures that they only have the necessary permissions to manage and update student grades. This also prevents accidental or unauthorized modifications that could compromise the integrity of the grading system.

Inside each of the sub-directories (e.g. assignment1, midterm, lab3) there are student directories (000->093) as well as questions.pdf and solutions.pdf files. These two files are owned by root and group owned by lecturer.

```
drwxrwxr-x  96 root lecturer   1960 Apr 14 14:59 .
drwxrwxr-x  11 root lecturer    240 Apr 14 14:59 ..
-rwxrw-r--   1 root lecturer      0 Apr 14 14:59 questions.pdf
-rwxrw----+  1 root lecturer      0 Apr 14 14:59 solutions.pdf
drwxrwx---+  2 root student000   60 Apr 14 14:59 student000
drwxrwx---+  2 root student001   60 Apr 14 14:59 student001
drwxrwx---+  2 root student002   60 Apr 14 14:59 student002
drwxrwx---+  2 root student003   60 Apr 14 14:59 student003
```

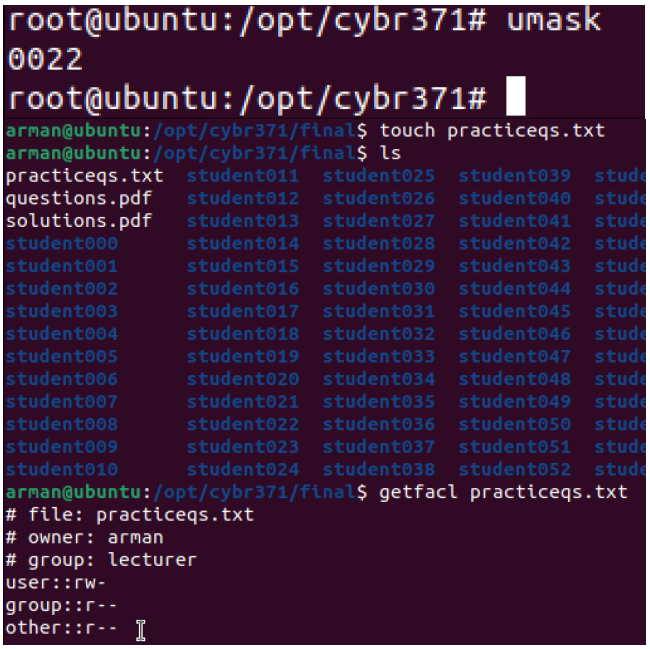
The questions.pdf allows the lecturer to have rw- access to ensure that they can construct appropriate questions for their respective assessment (in this case assignment1). Granting the lectures group read and write access only ensures that they have the necessary permissions to manage and construct the questions.pdf file while preventing accidental/unauthorized modifications that would fail the layout of the system. This also has read permissions for other groups (tutors, students) to view the questions file so that tutors get a grasp on what they will be marking, and students can do their assessment.

The solutions.pdf allows the lecturer to have read and write access so they can construct and modify the solutions for their respective assessment. Granting the lectures read and write access allows the group to have only the necessary permissions to manage this file without making unnecessary modifications and removing the file itself. setfacl is used on this file to grant tutors read access to the file so they can grade that assessment. Limiting the tutor's permissions to read access only allows them to not make written changes or deletions of the file which they should not be able to do. The student's group is not allowed to have any read, write, or execute permissions on this file to keep the integrity of the assessments and prevent cheating.

The student directories in each assessment directory are owned by root. Assigning root ensures admins have full control over the directories which allows unauthorized modifications or access by users who do not have access to that directory. Each student directory is group owned by its respective student group (student000) with rwx permissions. This allows students to 'upload' their answers to this directory to submit their assessment. This gives them the necessary permissions to have control over the directory. I have used setfacl to grant read and execute access to lecturers and tutors to ensure that they can have access to that directory to read student submissions. Granting no access to other ensures that other students cannot access that directory to have a look at the other student's answers. Both lectures and tutors should have access to that directory so they can mark the respective student's assessment so they can then grade that student in the grades.xlsx file.

Inside each student directory (e.g. student000) is where each student uploads/modifies their answers.docx file for each assessment. It is owned and group-owned by the respective student for their privacy and control. This also ensures that each student has full control over their answers as well as being able to manage and modify their files without relying on root/admin intervention. By students owning their answers, they are responsible for their work which makes them accountable for their submissions. Other permissions (lecturer, tutor) are set to read-only which ensures that only the specific student who owns the file can modify its contents and prevents tampering.

A umask value of 0022 sets the default permissions for created files to be readable and writeable by the owner, and readable by the members of the owner group and others. Since we have set permissions on who can and can't read and execute the setup shell script in the /opt directory through running the setup script we are limiting the permissions and access to the script so it cannot be run by other groups. Have also limited access permissions for certain files and directories throughout the script, for example the solutions.pdf is unable to be read by a Student, however a tutor should have access when marking. It also prevents newly created files from being executable. A umask of 022 is also often good in an educational system where lecturers need to share files with instructors and students. A umask of 0022 gives the default permissions of 644 (rw-r—r--) for files and 755 (rwx-rx-rx) for directories which ensures that students, tutors, and lecturers can access directories and read files created my lecturers and tutors.



For example, if a past exam was uploaded to the final directory, it will enable students to have permissions on that file to view it, while also not having permissions for them to tamper with the file.

CASE STUDY 2

Q.7

	register-patient.sh	**check-medication.sh**	**patients**	**patients/RickSanchez1950**	**patients/SummerSmith2007**
All Doctors	r-x	-	rwx	-	-
drloun	r-x	-	rwx	-	rw
drstethosc	r-x	-	rwx	rw	rw
All Nurses	-	rwx (sudo)	-	-	-
others	-	-	-	-	-

Q.8.

The filesystem is set up from the /opt directory using the setup-clinic.sh script. This script creates all necessary users, groups, and directories for WellingtonClinic and sets appropriate ACLs. When run it creates an administrator user under group sudo which it then assigns the user the appropriate permissions for control over scripts and directories. On running it sets the administrator to be the owner and group owner of the setup-clinic.sh script with read, write, and execute permissions and other with no permissions. This is so that the other users/groups (Doctor and Nurse) cannot run and write to the script as that would not be integral to access control across.

```
root@ubuntu:/opt# ls -al
total 12
drwxr-xr-x 1 root          root    120 Apr 14 21:51 .
drwxr-xr-x 1 root          root    260 Apr 14 08:36 ..
drwxr-x--- 3 administrator Doctor   60 Apr 14 21:26 WellingtonClinic
-rwxrwx--- 1 administrator sudo   1244 Apr 14 12:00 check-medication.sh
-rwxr-x--- 1 administrator Doctor 1563 Apr 14 21:51 register-patient.sh
-rwxrwx--- 1 administrator sudo   1369 Apr 14 21:25 setup-clinic.sh
```

Setup-clinic.sh also sets permissions for running the other two scripts: check-medication.sh and register-patient.sh. For register-patient.sh only members of Doctor should be able to set up a new patient. To achieve this, I have set the Doctor group to be the group owner of the register-patient script with read-and-execute access. This allows each doctor user to run the script effectively to create a new patient while also denying access to the Nurse group by not setting other permissions. This enables the least privilege to create a patient. Check-medication.sh is owned by the administrator and group owned by sudo. The reason for this is that the Nurse group should not be allowed to read/access an entire patient file. We achieved them not having access by removing Nurse access to the WellingtonClinic directory. The way they read the patient file is by running the check-medication.sh file as sudo user. We set Nurses to be able to run the check-medication.sh file with elevated sudo privileges only when running the check-medication.sh script so they will fully have read, write execute access across the WellingtonClinic. This means that there does not need to be any ACLs set for check-medication.sh.

For the WellingtonClinic directory, the administrator has owner and group owner of the directory with read, write, and execute permissions. This ensures that the administrator has overall control of the filesystem while being able to carry out necessary maintenance if needed. I have granted other execute access to the WellingtonClinic directory. This allows doctor and nurse users to go inside the WellingtonClinic directory.

The patient's directory inside WellingtonClinic is owned by the administrator with full read, write, execute permissions and Doctor with read, write, execute permissions. The reason for giving full access for doctors to be able to read, write, and execute to the directory is so that when they are registering new patients by running the register-patient.sh script they can write their patient file into this directory. There are no permissions given for others as nurses should not be able to access a patients entire file so there is no need to give permissions to others..... 'it follows least privilege'.

```
root@ubuntu:/opt# getfacl WellingtonClinic
# file: WellingtonClinic
# owner: administrator
# group: sudo
user::rwx
group::rwx
other::--x

root@ubuntu:/opt# cd WellingtonClinic
root@ubuntu:/opt/WellingtonClinic# getfacl patients
# file: patients
# owner: administrator
# group: Doctor
user::rwx
group::rwx
other::---

root@ubuntu:/opt# su drloun
drloun@ubuntu:/opt$ ./register-patient.sh
User is a Doctor
Enter Patient Information:
First name: Summer
Last name: Smith
Year of birth: 2007
Specify a doctors (~primary,#secondary): ~drloun,#drstethosc
Doctors are doctors
Doctors are doctors
# file: WellingtonClinic/patients/SummerSmith2007.txt
# owner: drloun
# group: Doctor
user::rw-
user:drstethosc:rw-
group::---
mask::rw-
other::---

File created at WellingtonClinic/patients/SummerSmith2007.txt
drloun@ubuntu:/opt$ exit
exit
root@ubuntu:/opt# su drstethosc
drstethosc@ubuntu:/opt$ ./register-patient.sh
User is a Doctor
Enter Patient Information:
First name: Rick
Last name: Sanches
Year of birth: 1950
Specify a doctors (~primary,#secondary): ~drstethosc
Doctors are doctors
# file: WellingtonClinic/patients/RickSanches1950.txt
# owner: drstethosc
# group: Doctor
user::rw-
group::---
other::---
```

To run the register-patient.sh directory it can only be run by a Doctor (Nurses do not have read, execute access to run). The doctor then specifies the appropriate information in constructing a new patient file which includes first and last name and birth year. They also specify the doctors for this patient. A patient can have a primary doctor which is the first doctor written when specified, as well as a second doctor. A patient can have multiple second doctors as well. From here the register-patient.sh script grants the primary doctor ownership of the patient file with read and write permissions as well as grants any secondary doctor read and write access. The group owner of the file is set to doctors, but it is set with no permissions as only doctors that are specified as primary and secondary can access that script. Other also have no permissions, which means that even though Nurses currently have no permissions to be able to access the patient directory, they will have no permissions to read the files to access an entire patient script. So, in this case; drloun and drstethosc are both able to access Summer Smith's patient file as they are both primary and secondary doctors to that patient, however for Rick Sanchez patient file only drstethosc can access it (drloun cannot read the file or write to it).

```
drbeas@ubuntu:/opt$ sudo ./check-medication.sh
Enter what patient you want to access
First name: Summer
Last name: Smith
Year of birth: 2007
Patient             Primary Doctor          Secondary Doctor
Summer Smith        Dr Lou Ngevity          Dr Stethos Cope

Date of Visit        Attended Doctor      Medication          Dosage

7/1/2024             drloun               adventurafil        qd
1/2/2024             drstethosc           scratchacine        prn
10/2/2024            drloun               placebazole         pc
```

From here a Nurse can access part of the patient directory by running the check-medication.sh using sudo permissions which would grant them full read, write, and execute access, enabling them to access the WellingtonClinic/patients directory to read patient files. From here the nurse provides patient information (first and last name, birth year) which will then output the necessary information from the patient file (visit date, attended doctor, meds, dosage). After running the script, the Nurse will be able to run it again with sudo privileges, but it will not have sudo permissions to then go and access files and directories it does not have permissions for.

```
root@ubuntu:/opt# ls
WellingtonClinic  check-medication.sh
root@ubuntu:/opt# umask
0022
root@ubuntu:/opt#
```

A umask value of 022 sets the default permissions for created files to be readable and writeable by the owner, and readable by the members of the owner group and others. However, it prevents newly created files from being executable. Since we have set permissions on who can and can't read and execute the existing shell scripts in the /opt directory through the setup-clinic.sh script we are limiting the permissions and access to the other shell scripts in the /opt directory. We have also limited access permissions for WellingtonClinic which would prevent unauthorized users that aren't given specified permissions from being able to read files that ACLs have not set for. A umask of 022 is also often good in a filesystem system where groups need to share files and directories in a workplace. A umask of 022 gives the default permissions of 644 (rw-r—r--) which ensures that groups doctors and nurses can access and read files necessary files for the clinic, while others that should be protected, have been specified with the necessary permissions for security.

Reference Declaration

- ChatGPT (helped clean up issues in code such as error checks in register-patient e.g. for loop to check patient is not doctor or nurse as well as checks if user was part of a group)
- https://www.baeldung.com/linux/edit-etc-sudoers-using-script (helped get understanding how to grant group limited sudo access for a group)
- https://stackoverflow.com/questions/16675179/how-to-use-sed-to-extract-substring (help to understand how sed works to extract strings. E.g. extracting doctors in register-patients.sh)
- https://stackoverflow.com/questions/28231360/command-sed-using-g (help to understand how to appropriately use sed. Used when extracting doctors.
- https://linuxhint.com/bash_tr_command/ (helped gain understanding of tr command when extracting vars)
- https://prathapreddy-mudium.medium.com/basic-usage-of-awk-command-ea564f2400fe (how to use awk to print lines from file)
- https://stackoverflow.com/questions/50707691/awk-adding-specific-number-of-spaces-between-columns (spacing between information when using awk)
- https://www.quora.com/How-do-you-use-the-cut-command-in-Bash (understanding of cut for when getting certain var in patient file for check-medication.sh)