Cam Olssen
300492582

2. Access matrix

	Register-patient.sh	Searchpatient.sh	Check-medication.sh	Patients folder	Wellington Clinic folder	MasoodMansoori2001	LanceBourne1970
All Doctors	r-x	r-x	---	rwx	r-x	---	---
DrMaryT	---	---	---	---	---	r--	r--
DrMandyS	---	---	---	---	---	---	r--
DrEliM	---	---	---	---	---	---	---
All Nurses	---	---	r-x	r-x	r-x	r--	r--

3. Commands

searchpatient.sh:

sudo chown BenM searchpatient.sh
sudo chgrp doctors searchpatient.sh
sudo chmod 750 searchpatient.sh
ls -l output

```
osboxes@osboxes:~/Desktop/scripts$ ls -l searchpatient.sh
-rwxr-x--- 1 BenM doctors 469 Apr 17 20:34 searchpatient.sh
```

getfacl output

```
osboxes@osboxes:~/Desktop/scripts$ getfacl searchpatient.sh
# file: searchpatient.sh
# owner: BenM
# group: doctors
user::rwx
group::r-x
other::---
```

check-medication.sh

sudo chown BenM check-medication.sh
sudo chgrp nurses check-medication.sh
sudo chmod 750 check-medication.sh
ls -l output

```
osboxes@osboxes:~/Desktop/scripts$ ls -l check-medication.sh
-rwxr-x--- 1 BenM nurses 718 Apr 17 19:45 check-medication.sh
```

getfacl output

```
osboxes@osboxes:~/Desktop/scripts$ getfacl check-medication.sh
# file: check-medication.sh
# owner: BenM
# group: nurses
user::rwx
group::r-x
other::---
```

Patients folder

Cam Olssen
300492582

sudo chown BenM patients
sudo chgrp admins patients
sudo chmod 740 patients
setfacl -Rdm g:doctors:rwx patients
setfacl -Rdm g:nurses:r-x patients
setfacl -Rm g:nurses:r-x patients
setfacl -Rm g:doctors:rwx patients
ls-l output

```
osboxes@osboxes:/opt/WellingtonClinic$ ls -l patients
total 16
-rw-rw----+ 1 BenM admins  51 Apr 17 20:37 JohnPatient1999.txt
-rwxrwxr--+ 1 BenM admins 205 Apr 15 19:05 LanceBourne1970.txt
-r-xrwx---+ 1 BenM admins 115 Apr 15 19:04 MasoodMansoori2001.txt
-rw-rwx---+ 1 BenM admins  40 Apr 15 21:06 TestTestson1.txt
```

getfacl output

```
osboxes@osboxes:/opt/WellingtonClinic$ getfacl patients
# file: patients
# owner: BenM
# group: admins
user::rwx
group::r--
group:doctors:rwx
group:nurses:r-x
mask::rwx
other::---
default:user::rwx
default:group::r--
default:group:doctors:rwx
default:group:nurses:r-x
default:mask::rwx
default:other::---
```

MasoodMansoori2001 file
setfacl -Rdm g:doctors:rwx patients
setfacl -Rdm g:nurses:r-x patients
ls -l output

```
osboxes@osboxes:/opt/WellingtonClinic/patients$ ls -l MasoodMansoori2001.txt
-r-xrwx---+ 1 BenM admins 115 Apr 15 19:04 MasoodMansoori2001.txt
```

getfacl output

```
osboxes@osboxes:/opt/WellingtonClinic/patients$ getfacl MasoodMansoori2001.txt
# file: MasoodMansoori2001.txt
# owner: BenM
# group: admins
user::r-x
group::rwx
group:doctors:rwx
group:nurses:r-x
mask::rwx
other::---
```

LanceBourne1970 file
setfacl -Rdm g:doctors:rwx patients
setfacl -Rdm g:nurses:r-x patients
ls -l output

```
osboxes@osboxes:/opt/WellingtonClinic/patients$ ls -l LanceBourne1970.txt
-rwxrwxr--+ 1 BenM admins 205 Apr 15 19:05 LanceBourne1970.txt
```

Cam Olssen
300492582

getfacl output

```
osboxes@osboxes:/opt/WellingtonClinic/patients$ getfacl LanceBourne1970.txt
# file: LanceBourne1970.txt
# owner: BenM
# group: admins
user::rwx
group::rwx
group:doctors:rwx
group:nurses:r-x
mask::rwx
other::r--
```

7. Patient information stored in the patients folder is meant to be confidential. However, due to the specified requirements of the file structure, there are several key flaws in the security of patient data which could be exploited by a threat agent.

The first of these is that doctors have read, write and execute permissions within the patients folder. This is a problem, as it means that they have full power to access and edit patient files, including those of patients who they are not assigned to. It is necessary to give doctors these permissions under the current structure, however, as register-patient.sh requires write permissions on patients in order to create the patient file, and read and execute permissions are required in order for doctor-run scripts such as searchpatient.sh to work.

This means that, if a threat agent were to gain access to a doctor account, they could edit any patient data to say whatever they wanted, as well as accessing it for themselves.

This security flaw could be resolved by not giving doctors these permissions and using setuid on the individual scripts instead. However, most Linux distributions, Ubuntu included, do not allow setuid on Bash/shell scripts. This is for security reasons, as it could allow an attacker to gain access to a shell with permissions beyond what they should normally have.

A potential resolution to this security flaw would be to call the shell script from a program which can use the setuid bit. This will allow for reducing permissions of doctors on the patients directory, which will prevent them from accessing data that they are not supposed to under the assignment brief. This also means that, should a threat agent have access to a doctor account they would not be able to change data for existing patients directly, which would prevent them from causing as much damage. They would also be unable to access patient data beyond what can be accessed from the scripts.

Nurses having read and execute permissions on patients is also a security flaw. The reason why they have this in the current structure is because it is required for checkmedication.sh to run. However, this also grants them access to view all patient data, which they do not need and thus should not have under the principle of least privilege. If a threat agent gained access to a nurse account, they would be able to access privileged patient data, violating the clinic's confidentiality.

While using setuid would be ideal, it is disabled for Bash/shell scripts as detailed earlier. As such, a solution to this vulnerability is the same as for the previous one – implementing a wrapper program that can use the setuid bit and run checkmedication.sh with elevated privileges, allowing the nurse to access the patient data only through the program.

Another vulnerability is in the create-directory-staff.sh script. Due to the specifications of the assignment, all user passwords are set inside the script in plaintext. This means that if an attacker

were able to gain read access on that script, they would be able to access the passwords for all users on the network, which is a massive security breach.

A potential solution to this is through setting expiry dates for the passwords through use of the chage command, requiring the user to change their password shortly after the system is created. This means that, in the event an attacker gains access to create-directory-staff.sh, the passwords stored inside will not be of use to them due to the users having been forced to change them.

8. I have written wrapper programs in C for checkmedication.sh, registerpatient.sh and searchpatient.sh, using setuid to run them with elevated privileges. I elected not to implement the password expiry solution as it would slow down marking of the assignment. These programs do work for running the script, however I have not fully implemented the solution so that the scripts can be tested according to the assignment brief.

These programs work as intended, and prevent nurses and doctors from gaining access to patient data that they do not need in accordance with the principle of least privilege. This does not eliminate the risk of a threat agent gaining access to a nurse or doctor's account and using it to access confidential patient information – however, it does mitigate this risk by limiting the patient information that they can access substantially.