

Systems Software Report CA1

TU857

BSc in Computer Science

**Omair Duadu**

**C18322011**

School of Computer Science

TU Dublin – City Campus

**02/11/2021**

**Table of Contents**

*Functionality Checklist* 3

*Feature 1 - System Architecture including makefile* 4

*Feature 2 - Daemon (Setup/ Initialisation/ Management)* 4

*Feature 3 - Daemon (Implementation)* 4

*Feature 4 - Backup Functionality* 5

*Feature 5 - Transfer Functionality* 5

*Feature 6 - Lockdown directories for Backup / Transfer* 5

*Feature 7 – Process management and IPC* 5

*Feature 8 - Logging and Error Logging* 5

*Conclusion* 6

# *Functionality Checklist*

|  |  |  |
| --- | --- | --- |
| ***Feature*** | ***Description*** | ***Implemented*** |
| F1 | System Architecture including makefile | Yes |
| F2 | Daemon (Setup/Initialisation/Management) | Yes |
| F3 | Daemon (Implementation) | Yes |
| F4 | Backup Functionality | Yes |
| F5 | Transfer Functionality | Yes |
| F6 | Lockdown folder for Backup / Transfer | Yes |
| F7 | Process management and IPC | Yes |
| F8 | Logging and Error Logging | Yes |

Have you included a video demo as part of the assignment: Yes

Link to Video: <https://youtu.be/plskpMMflJ4>

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:

Omair Duadu

02/11/2021

# *Feature 1 - System Architecture including makefile*

The system I am making has 2 core functions which are transferring files from the Shared directory to the Dashboard directory, and then performing a backup of the Dashboard.

The daemon is configured to run at 11.30pm every night so that there is minimal impact on the performance because will the transfer/backup is being completed access to these directories will be lost. The operating system I am using is a Debian Linux, which uses systemd to initialise this daemon.

The detailed description of how the daemon works is that it uses a Singleton Pattern, which creates a child process, where an orphan is created and elevated to session leader, this is important so that it loses controlling TTY. From there umask() to gain all permissions required.

The next bit which is highly import is the logic runs in an infinite loop, it contains the lockShared(), transfer(), and backup() functionality.

A picture containing graphical user interface

Description automatically generated

The code files and log folder are all present in one directory, I have applied Separation of Concerns by making different files for all the functions rather than placing them all in one big code file.

Graphical user interface, application, Teams

Description automatically generated

All the different directories are placed in the Plant directory like you can see above, this includes the Shared, Dashboard and Backup folders.

Text

Description automatically generated

My makefile is able to quickly compile and run all the code I need which includes the object & header files with a single command: make!

Make clean also gives me the ability to remove all the object files to recompile

Architecture Diagram.

Service

mainProgram

backup

lockShared

transfer

# *Feature 2 - Daemon (Setup/ Initialisation/ Management)*

From the very beginning the first thing in the system is a systemd service called myDaemon.service. Systemd has many great functions and the 5 main ones for my use case are start, stop, status, enable and disable.

Sudo systemctl start/stop myDaemon.service can start/stop the service at will.

Sudo systemctl enable/disable myDaemon.service can start/stop the service at boot time, this essential automates so that even if the system shuts down it will still work the next time it turns on.

The myDaemon.service and similar service files are stored in the /etc/systemd/system directory. There are also other locations for service files with different purposes.

The target for the service is stored in the /usr/bin/ directory.

Text

Description automatically generated

Part of my management and initialisation is creating a pathFile.h which holds defined paths which is used all across the different functions for uses.

Text

Description automatically generated

# *Feature 3 - Daemon (Implementation)*

Text

Description automatically generated

Including all the libraries and header files of the functions needed.

Text

Description automatically generated

This is how the time is set for when the daemon will run.

PROCESS MANAGEMENT

Text

Description automatically generated

From the parent process create a child the orphan process.



Make the orphan the new session leader

A screenshot of a computer

Description automatically generated with medium confidence

Give the process all permissions that it requires

Text

Description automatically generated

Go to the root directory to prevent problems.

Text

Description automatically generated

The last part of the Daemon is where it is ready to call the different functions.

There is other functionality in the daemon which is covered elsewhere.

*Feature 4 - Backup Functionality*

Detailed description of the backup implementation

The backup function is called only after lockShared() which changes the permissions, this is covered in more detail in the *Feature 6*

The backup file code contains all the libraries and header files that the program needs are included.

All system commands are made through system() calls.

The backup has all the necessary logging through syslog.

A new directory is created for every day in the backup.zip folder, using the mkdir command and BACKUP variable.



The backup is made by performing an “rsync -a” of the Dashboard directory to the backup location. The logging is created for a failure and success of the function.

After the backup the permissions are restored for all users.

Text

Description automatically generated

# *Feature 5 - Transfer Functionality*

Detailed description of the transfer implementation

The backup function is called only after lockShared() which changes the permissions, this is covered in more detail in the *Feature 6*

The backup file code contains all the libraries and header files that the program needs are included.

All system commands are made through system() calls.

The backup has all the necessary logging through syslog.

The transfer functionality is very similar to the backup, except that the contents of Shared are moved to the Dashboard, and no new directory is created.

Part of transfer() is that it checks for if the managers have uploaded a new file the day, and this is done by making an array of path variables and comparing that to what is found in the Shared directory.

Text

Description automatically generated

# *Feature 6 - Lockdown directories for Backup / Transfer*

Detailed description of the lockdown functionality/implementation

The lockdown of permissions is the first function that is triggered in the main program, and also triggered in the backup function.

2 parameters need to be passed to the function which are mode[] for the permission level and path[] for the path where the permissions will be changed.

Chmod is the command for changing permission levels.

Chmod 0000 closes all permissions to all users except root, Chmod777 then will reopen all permissions.

Text

Description automatically generated

# *Feature 7 – Process management and IPC*

Detailed description of how child processes communicate success/failure of tack to be completed to parent process etc….

I have achieved process management by being able to fork process and be able to observe the result of the process wherever it acts throughout the system.

This is achieved through applying abundant logging through syslog which is a very efficient system. The syslog was saved to syslog.txt.

Text

Description automatically generated

Text

Description automatically generated with medium confidence

# *Feature 8 - Logging and Error Logging*

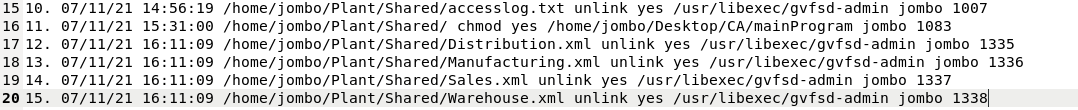
Detailed description of the error and logging functionality included in the code solution.

Graphical user interface, application

Description automatically generated

The logging is stored in one directory /logs/ accesslog.txt holds the user access logs, syslog.txt holds the syslog details.

Accesslog.txt looks like this its displays actions taken, user, and time.



Syslogs.txt syslog looks like this it displays time user, process, and the syslog message

Text

Description automatically generated with medium confidence

Syslog messages are generated through functions





Syslog is opened through the openlog.

/usr/sbin/auditctl records user actions into accesslog.txt from where they can be viewed easily

Text

Description automatically generated

# *Conclusion*

Summary of the implementation and achievement

The Daemon is built like a lego set, each part which is created is interconnected with other pieces.

The main daemon has been added to the service files, giving many amazing functionality which is provided by systemd.

When the daemon it forks a process, it also has a countdown function which sets when the transfer and backup will occur. The forked orphan process will perform the permission locking, backup and transfer.

Logging is also present through logging for user actions by using auditd and syslogs which are activated throughout the functions.