Instituto Tecnológico de Costa Rica

Ingeniería en Computación

Curso

Principios de Sistemas Operativos

Proyecto 2 - QRFS

Estudiantes:

Joan Sánchez 2015123867
Rodrigo Espinach 2014055978

Profesor

Kevin Moraga

22 de junio de 2022

1. Introducción

QRFS es un sistema de archivos que reside en el espacio de usuario. Este sistema de archivos tiene como objetivo utilizar los imágenes basadas en código QR para almacenar archivos. Para QRFS el espacio físico donde se almacena su información y toda su estructura se encuentra en todas las imágenes QR, donde cada imagen representa un bloque del FS.
Como parte del segundo proyecto del presente curso, se planea desarrollar un sistema de archivos con los métodos tipicos getattr, create, open, read, write, rename, mkdir, readdir, opendir, rmdir, statfs, fsync y access. Esta se implementará por medio del lenguaje de programación Rust. Adicional a esto se realizará la implementación de 3 archivos, mount.qrfs que seria el archivo de montaje del sistema de archivos, el archivo fsck.qrfs, el cual, se utiliza para verificar el estado del sistema y finalmente el archivo mkfs.qrfs, este se encargará de crear un nuevo sistema de archivos.

2. Ambiente de Desarrollo

Para este proyecto se utilizó un ambiente de desarrollo que se compone de Ubuntu 20.04 corriendo en WSL 2 para Windows, una máquina virtual con la misma versión de Ubuntu, Visual Studio Code como editor de texto y cargo como administrador de paquetes. Además, para la comunicación entre el equipo de trabajo se utilizó Telegram y Discord. Finalmente, para administrar el control de versiones del programa se utilizo github.

3. Estructuras de Datos Usadas y Funciones

- Archivo mkfs-QRFS
  - Inode: unidad que movera nuestro fs
    - changeName(value: String): Función utilizada para cambiar el nombre de Inode
    - addReference(refValue: usize): Agrega una referencia a Inode
    - deleteReference(refValue: usize): Elimia una referencia del Inode
  - memBlock: estructura para guardar el contenido de cada Inode creado
    - addData(data: u8): Agrega una referencia de el mismo
    - deleteData(data: u8): Elimina una referencia de el mismo
  - Disk: estructura para guardar todos los datos del Inode
    - new(path:String , diskPath:String): Crea un disco e Inode raíz
    - returnNextInode(): retorna el siguiente Inode
    - writeInode(inod:Inode): Agrega un Inode al super bloque
    - removeInode(inode: u64): Elimina el Inode disponible
    - clearReference(inod: u64, refValue: usize): Elimina una referencia de un Inode en el super bloque
    - addReference(inod: u64, refValue: usize): Añade una referencia de un Inode en el super bloque
    - getInode(inod: u64): Obtiene el Inode
    - getMutInode(inod: u64): Obtiene el Inode mutable
    - findInodeByName(parentInode: u64, name: str): Busca en base al Inode padre el hijo que tenga el nombre enviado por parametro.

- addDataInode(inod:u64,data:u8): Agrega datos al bloque de memoria asociado al Inode
- writeContent(referenceInode: u64, content: Vec¡u8¿): Agrega un arreglo de bites dentro de un inode
- deleteDataInode(inod:u64,data: u8): Elimina la data el bloque de memoria asociado al inod
- getBytesContent(inod: u64): Obtiene un arreglo de contenidos asociado al inod

- Filesystem: estructura del sistema de archivos, solo posee un disco
  - new(rootPath:String, diskPath:String): Crea un nuevo disco
  - getDisk(): Obtiene el disco
  - setDisk(newDisk:Disk): Inserta un nuevo disco
  - saveFileSystem(): Guarda el disco como un QR
  - looup(req: Request, parent: u64, name: OsStr, reply: ReplyEntry): Busca un directorio y obtiene sus atributos
  - getattr(req: Request, inod: u64, reply: ReplyAttr): Busca el inode asignado al inod y devuelve sus atributos
  - create(req: Request, parent: u64, name: OsStr, mode: u32, flags: u32, reply: ReplyCreate): Crea un archivo en el padre pasado por parametro
  - open():
  - read(req: Request, inod: u64, fh: u64, offset: i64, size: u32, reply: ReplyData): Busca el bloque de memoria asignado al inod y muestra su contenido
  - write(req: Request, inod: u64, fh: u64, offset: i64, data: [u8], flags: u32, reply: ReplyWrite): Escribe dentro de un archivo en base al inod pasado
  - rename(req:Request, parent:u64, name:OsStr, newparent: u64, newname:OsStr, reply:ReplyEmpty): cambia de nombre a un archivo mediante el padre que se envia por parametros
  - mkdir(req: Request, parent: u64, name: OsStr, mode: u32, reply: ReplyEntry): Crea un nuevo directorio asignando un Inode
  - readdir(req: Request, inod: u64, fh: u64, offset: i64, mut reply: ReplyDirectory): Lee el directorio
  - opendir(req: Request, inod: u64, flags: u32, reply: ReplyOpen): Abre un directorio
  - rmdir(req: Request, parent: u64, name: OsStr, reply: ReplyEmpty): Elimina un directorio en base al nombre
  - statfs(req: Request, ino: u64, reply: ReplyStatfs): Devuelve las estadistcas del filesystem
  - fsync(req: Request, inod: u64, fh: u64, datasync: bool, reply: ReplyEmpty): Vacia los datos de disco y del usuario
  - access(req: Request, ino: u64, mask: u32, reply: ReplyEmpty): Revisa el acceso de los permisos

- Archivo mount-QRFS
  - mount-qrfs(): Se encarga de crear un nuevo sistema de archivos QRFS.

- Archivo fsck-QRFS
  - checkConsistence(fs:fileSystem): Verifica la consistencia del sistema de archivos

- Archivo sesInformation
  - Definimos datos necesarios para la implementación del sistema

4. Instrucciones para ejecutar el programa

- Primeramente se debe posicionar a través de una terminal, en el directorio que contiene el proyecto.
- Una vez allí, se debe escribir 'cargo build' para que el adminstrador de paquetes compile los archivos correspondientes.
- Seguidamente escribir el comando 'cargo run [Lugar donde se encuentra el disco] [Lugar donde estará el sistema de archivos]' para ejecutar el programa.

  Ejemplo visual:
  Posicionarse en el directorio del proyecto:

```
tinky-winky@tinky-winky:~/Documents/Proyecto02SO-QRFS/proyecto02SO$
```

  Ejecutar cargo build:

```
tinky-winky@tinky-winky:~/Documents/Proyecto02SO-QRFS/proyecto02SO$ cargo build
```

  Ejecutar cargo run seguido de la dirección del disco y la dirección donde estará el sistema de archivos:

```
tinky-winky@tinky-winky:~/Documents/Proyecto02SO-QRFS/proyecto02SO$
cargo run /home/tinky-winky/Documents/Proyecto02SO-QRFS/proyecto02SO/src/Storage
/home/tinky-winky/pruebas_disco
```

5. Actividades Realizadas por los estudiante

- Timesheet de Rodrigo

Fecha	Actividad realizada	Tiempo Invertido
30/06/2022	Se trabaja en el kick off del proyecto	2h
4/06/2022	Investigación sobre FUSE y sus funciones	6h
7/06/2022	Investigación sobre filesystem	5h
9/06/2022	Investigación sobre i-nodos	4h
11/06/2022	Investigación sobre cifrado de listas/archivos	4h
13/06/2022	Planteamiento del diseño del FS y comienzo del mkfs.qrfs	9h
16/06/2022	Intento de Re-Implementación de las funciones de FUSE	4h
18/06/2022	Corrección de errores y mejoras al mkfs.qrfs	4h
19/06/2022	Se termina la implementación del mkfs.qrfs Se intenta implementar el fsck.qrfs y el mount.qrfs	8h
20/06/2022	Se corrigen errores y se finaliza el fsck.qrfs y el mount.qrfs	10h
21/06/2022	Se corren pruebas finales, se solucionan últimos errores y se trabaja en la documentación final	12h

Total de horas invertidas de Rodrigo: 68.

- Timesheet de Joan

Fecha	Actividad realizada	Tiempo Invertido
30/06/2022	Se trabaja en el kick off del proyecto	2h
4/06/2022	Investigación sobre FUSE y sus funciones	6h
7/06/2022	Investigación sobre filesystem	5h
9/06/2022	Investigación sobre i-nodos	4h
11/06/2022	Investigación sobre cifrado de listas/archivos	4h
13/06/2022	Planteamiento del diseño del FS y comienzo del mkfs.qrfs	9h
16/06/2022	Intento de Re-Implementación de las funciones de FUSE	4h
18/06/2022	Corrección de errores y mejoras al mkfs.qrfs	4h
19/06/2022	Se termina la implementación del mkfs.qrfs Se intenta implementar el fsck.qrfs y el mount.qrfs	8h
20/06/2022	Se investiga y se implementa el lector de QR para móviles	9h
21/06/2022	Se corren pruebas finales, se solucionan últimos errores y se trabaja en la documentación final	12h

Total de horas invertidas de Joan: 67.

El total de horas invertidas entre los dos estudiantes es de 135.

6. Evaluación

- QRFS:
- mkfs.qrfs: 11
- fsck.qrfs: 5
- mount.qrfs: 15
- Funciones de la biblioteca: 24
  - getattr 2
  - create 2
  - open 2
  - read 2
  - write 2
  - rename 2
  - mkdir 2
  - readdir 2
  - opendir 2
  - rmdir 2
  - statfs 2
  - fsync 0
  - access 2
- Documentación: 15
- Diseño del FS: 10
- Manejo de Fragmentación: 15

7. Autoevaluación

Al momento de entrega, el proyecto se encuentra casi completo. Los requerimientos del passphrase para firmar el FS y el sistema de cifrado no se lograron efectuar por falta tiempo, además no se logro implementar el metodo fsync.

Se considera que la carga que representaba la implementación completa del proyecto estuvo un poco superior al tiempo que ambos estudiantes disponían, ya que, al ser un curso de 4 creditos no se esperaba dedicar tanto tiempo mayororitariamente a la investigación. Como limitaciones se encontró que, a pesar de llevar todo el curso programando en rust, sigue siendo un lenguaje que cuesta un poco de adaptarse y requerimos un poco de tiempo al encontrarnos con errores desconocidos. Para este caso, otra limitante pero no tan marcada fue un poco la dificultad de encontrar información, esto porque se requiere conocer bien que se desea preguntar para encontrar algo de ayuda, sin embargo en comparación con el primer proyecto consideramos que no estuvo tan complejo.

- Joan:

- 5 Aprendizaje de mkfs
- 5 Aprendizaje de fsck.
- 5 Aprendizaje de mount.
- 5 Aprendizaje de implementacion de funciones.
- 5 Aprendizaje de Diseño de Filesystem.

- Rodrigo:
  - 5 Aprendizaje de mkfs
  - 5 Aprendizaje de fsck.
  - 5 Aprendizaje de mount.
  - 5 Aprendizaje de implementacion de funciones.
  - 5 Aprendizaje de Diseño de Filesystem.

- Registro de commits de Git

```
PS C:\Users\joans\Desktop\Proyecto02SO-QRFS> git log
commit 8329b7732b53a11e3d5c3955ad2cfc27a7c2e317 (HEAD -> main, origin/main, origin/HEAD)
commit 8329b7732b53a11e3d5c3955ad2cfc27a7c2e317 (HEAD -> main, origin/main, origin/HEAD)
Author: JASCH97 <joansc16@outlook.com>
Date:   Wed Jun 22 02:17:55 2022 -0600

    Se agrega la aplicacion para android capaz de leer y enviar codigos QR

commit 56dde1ec2dbad6a7ce2d1d24c2e7cafd9ecc3f2e
Author: Rodri2210eh <naliuska.espher@gmail.com>
Date:   Tue Jun 21 22:39:55 2022 -0600

    Agregando comentarios a las fn

commit 3f754a30fd941397d13fdefc86de365287c46b33
Author: JASCH97 <joansc16@outlook.com>
Date:   Tue Jun 21 21:23:29 2022 -0600

    se logra generar un qr que representa el sistema de archivos

commit ee0a4581809546fcfe5fd285ecbe590df309809e
Author: JASCH97 <joansc16@outlook.com>
Date:   Tue Jun 21 21:01:56 2022 -0600

    se corrigen mas errores y se logra correr correctamente el FS

commit ee2a364c723d99891f081dd1330c734cb58320ff
Author: JASCH97 <joansc16@outlook.com>
Date:   Tue Jun 21 19:30:27 2022 -0600

    se corrigen errores de nombres de variables y falta de dependencias

commit a96bab4175ea20a37ca41b34775a34c0a10201ee
Author: JASCH97 <joansc16@outlook.com>
Date:   Tue Jun 21 19:27:45 2022 -0600

    se corrigen errores de nombres de variables y falta de dependencias
```

```
commit 5be36daed6b2ffca67c07598a5a4e963bcff2536
Author: Rodri2210eh <naliuska.espher@gmail.com>
Date:   Tue Jun 21 16:45:46 2022 -0600

    solucionando errorcito

commit 7a474afa63799f0f2f71edde294ad46af1c98b49
Author: Rodri2210eh <naliuska.espher@gmail.com>
Date:   Tue Jun 21 16:43:48 2022 -0600

    Solucionando rename, y statfs

commit 3838a8fad9f053b5ae2a9d1ae27ced6e311bc1e9
Author: Rodri2210eh <naliuska.espher@gmail.com>
Date:   Tue Jun 21 15:16:02 2022 -0600

    modificando el nombre de los archivos

commit 86391d24ef31893841f19fcec0aa3e6d1a65d1bb
Author: Rodri2210eh <naliuska.espher@gmail.com>
Date:   Tue Jun 21 14:28:10 2022 -0600

    agregando estructura correcta para el proyecto falta los binarios

commit 971a570a45fd5b2a413680c9902aa6943440e5c3
Author: Rodri2210eh <naliuska.espher@gmail.com>
Date:   Mon Jun 20 16:31:35 2022 -0600

    soolucion errores 2.0

commit beaadcb919f3bb40d167e2b8bcae265fef9aedc1
Author: Rodri2210eh <naliuska.espher@gmail.com>
Date:   Mon Jun 20 16:19:29 2022 -0600

    primeras pruebas solucionando errores

commit 56869460cac048156ee4f51e211eac376d10c354
Author: Rodri2210eh <naliuska.espher@gmail.com>
Date:   Mon Jun 20 14:07:15 2022 -0600

    Agregando FS con fuse listo para pruebas

commit b9d33f3dca8b37eeb45c0ac5d5c8d6e9154456e8
Author: Rodri2210eh <naliuska.espher@gmail.com>
Date:   Mon May 30 19:06:53 2022 -0600

    Initial commit
~
```

8. Lecciones Aprendidas

Entre las lecciones aprendidas se pueden mencionar:

- Se logró entender el funcionamiento de fuse para poder implementar un FS.
- Se pudo entender al menos un poco el funcionamiento que tiene los Inode.
- Durante la investigación también se encontro como hacer un FS desde 0 sin usar fuse, por ende, logramos entender un poco más como funcionan estos.
- Se logro aprender como realizar un QRcode con rust.
- También se pudo trabajar más con rust lo cual hace que nuestra práctica y conocimiento en este lenguaje haya mejorado.

9. Bibliografía

- https://pike.lysator.liu.se/generated/manual/modref/ex/predef$3A$3A/Fuse/Operations/
- https://docs.rs/fuse/latest/fuse/
- https://docs.rs/fuse/latest/fuse/index.html
- https://zsiciarz.github.io/24daysofrust/book/vol1/day16.html
- https://www.cs.hmc.edu/ geoff/classes/hmc.cs135.201109/homework/fuse/fuse$doc.htmlfunction-purposes$
- https://github.com/zargony/fuse-rs
- https://docs.rs/fusemt/latest/fuse$_mt/https://www.geeksforgeeks.org/inode-in-operating-system/$
- https://www.hackplayers.com/2016/12/cifrar-y-descifrar-archivos-y-dirs-linux.html
- https://gustavopeiretti.com/java-codigos-barra-zxing/
- https://www.geeksforgeeks.org/how-to-generate-and-read-qr-code-with-java-using-zxing-library/
- https://www.javatpoint.com/how-to-send-email-in-android-using-intent