Tecnológico de Monterrey

Recuperación de Archivos - RAID 5

Equipo 10

Dael Chávez Ferreyra | A01771963
José Vicente Guadarrama Hernández | A01771927

25 de noviembre del 2024

Análisis y diseño de algoritmos avanzados

Profesor José María Aguilera Méndez

Introducción

El desarrollo de este último proyecto consiste en la simulación del algoritmo RAID, específicamente el algoritmo RAID 5, tomando el enfoque de recuperación de datos. Este algoritmo es originalmente aplicado para la recuperación de discos duros dentro del área de soporte y mantenimiento de bases de datos de las industrias. Sin embargo, para la aplicación hemos adaptado el algoritmo para que el mismo funcionara no sobre un conjunto de discos, sino sobre un conjunto de archivos de cualquier tipo, cumpliendo sobre un tipo de analogía en el que los archivos representan discos.

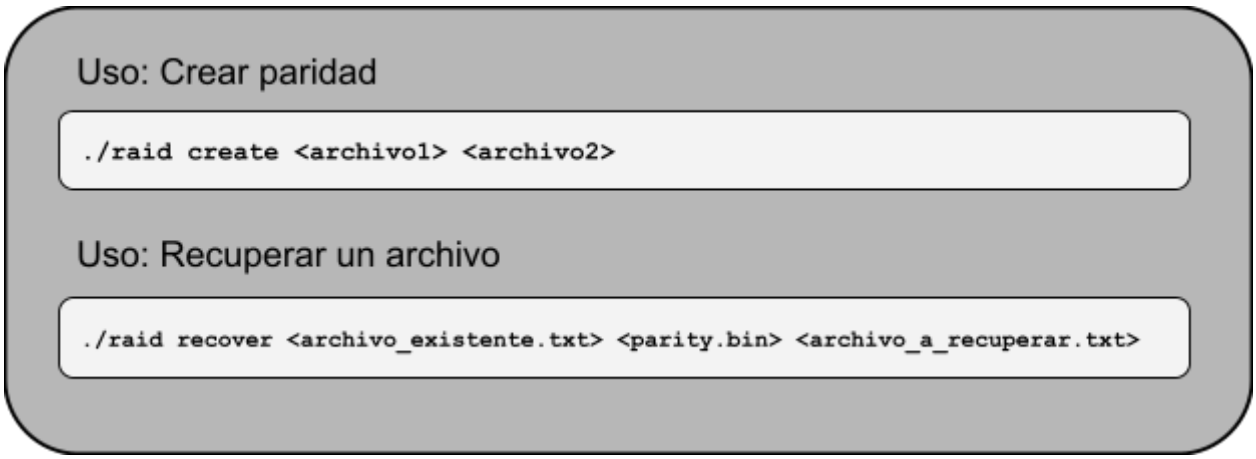
Desarrollo

Si bien existen diversos tipos de RAID hemos implementado el RAID 5, el cual está enfocado a la recuperación de datos por medio de una paridad, esta siendo un resultado de una operación XOR. El mismo algoritmo de RAID 5 ha servido como base para otros algoritmos como el RAID 6 o el RAID 5+0, por mencionar algunos ejemplos.

Lo que el programa hace es crear el archivo de paridad a partir de dos archivos, los cuales son dados por el usuario mediante un comando en consola, además mientras el programa se ejecuta se monitorean ambos archivos con el objetivo de identificar qué sucede con estos, ya sea que han sido modificados, o bien eliminados, permitiéndonos actualizar el archivo de paridad el cual será utilizado para recuperar información en caso de que esta se vea comprometida.

Adicionalmente, el programa hace la recuperación de información por medio de otro comando de consola, utilizando el archivo de paridad creado anteriormente y el archivo que no ha sufrido de alguna pérdida o modificación. Cabe mencionar, que el programa asume que el primer comando mencionado, es decir, el cual crea el archivo de paridad, ya ha sido ejecutado anteriormente.

Los comandos a ejecutar son:



Para darle mayor crédito al programa, existe la posibilidad de simular un error en el cual uno de nuestros archivos es modificado, reescribiendo un byte convirtiéndo el archivo como uno

diferente al que era originalmente, en otras palabras, corrompiéndolo. Para ello, el usuario puede ejecutar el comando directamente sobre la consola:

```
printf "\xFF" | dd of=<archivo> bs=1 seek=100 count=1 conv=notrunc
```

En este ejemplo, escribimos un caracter no válido sobre el archivo especificado en ‘of’, exactamente en el byte número 100 del archivo.

Para la correcta simulación de este error es necesario primero haber creado el archivo de paridad con el comando mostrado anteriormente, posterior a ello, podemos ejecutar este comando directamente sobre la CMD especificando el archivo objetivo y, si el usuario desea, modificar también el byte a escribir ya sea su valor o posición o ambos.

Una vez ejecutando el comando, podemos verificar los cambios en el archivo ejecutando:

Mostrar archivo modificado

```
cat <archivo_modificado>
```

Mostrar datos del archivo modificado

```
stat <archivo_modificado>
```

Aún si el archivo modificado resulta siendo de un tamaño diferente al archivo de paridad o el otro archivo, el programa será capaz de recuperar el archivo modificado gracias al algoritmo de RAID 5 y el archivo de paridad creado previamente, esto ejecutando el comando de recuperación dado previamente.

Si el usuario lo desea, podrá comprobar la recuperación de sus archivos utilizando los mismos comandos mostrados anteriormente.

Como otra funcionalidad del programa, este recuperará de manera automática un archivo que ha sido eliminado repentinamente, esto permitiendo al usuario recuperar su información sin necesidad de que este interactúe con el sistema y ofreciendo mayor seguridad de su información. Para esto no es necesaria la ejecución de otros comandos más que el comando que crea el archivo de paridad y el comando que recupera el archivo perdido.

Restricciones y consideraciones del programa

El programa ha sido desarrollado considerando varios aspectos que son de relevancia para que el mismo sea funcional y logre el objetivo exitosamente. El programa, siendo desarrollado dentro de un ambiente Linux, incluye algunas restricciones que deben ser consideradas para su ejecución como las dependencias, el ambiente de compilación y ejecución, entre otras.

Debido a que el programa incluye la librería de Inotify para el monitoreo de los archivos, la cual es propia de los ambientes de desarrollo de Linux, es estrictamente necesario contar con esta librería y un sistema operativo de desarrollo Linux. De igual manera, se requieren de dos archivos de exactamente el mismo tamaño y tipo, esto debido a las restricciones planteadas por el mismo algoritmo de RAID 5 en el cual nos basamos para el desarrollo.

Por lo anterior, también se asume que el usuario conoce cómo utilizar estos ambientes operativos, además, el usuario conoce qué archivos está manejando para la ejecución de este programa, asegurándose de que estos son del mismo tipo y tamaño.

Cabe destacar que, mientras el programa se ejecute en un ambiente Linux actualizado y con un compilador de GCC actualizado, y además se ejecute con dos archivos de mismo tipo y tamaño, el programa funcionará de manera exitosa para la creación de una paridad y la recuperación de información por medio de la paridad.

Recomendaciones

La correcta compilación y ejecución del programa es:

1. Compilar el programa principal:
   - `gcc raid.c -o raid`
2. Ejecutar el programa principal (crear el archivo paridad):
   - `./raid create <archivo1> <archivo2>`
3. Verificar la creación de paridad (opcional):
   - `ls`
4. Simular modificación o eliminación de un archivo:
   - `rm <archivo>`
   - `printf "\xFF" | dd of=<archivo> bs=1 seek=100 count=1 conv=notrunc`
5. Verificar cambios:
   - `ls`
   - `cat <archivo>`
   - `stat <archivo>`
6. Ejecutar recuperación:
   - `./raid recover <archivo_existente> <paridad> <archivo_a_recuperar>`
7. Verificar cambios:
   - `ls`
   - `cat <archivo>`
   - `stat <archivo>`