Sistemas Operativos I 2022
Trabajo Práctico Obligatorio 5

Objetivos

- Comprender de manera básica la relación entre bloques de disco y estructuras del Sistemas de Archivos.
- Acceder al contenido de un archivo utilizando funciones de la biblioteca de C básicas.

Referencias

[1] Tanenbaum, Bos – Modern Operating Systems - Prentice Hall; 4 edition (March 10, 2014) - ISBN-10: 013359162X
[2] Douglas Comer - Operating System Design - The Xinu Approach. CRC Press, 2015. ISBN : 9781498712439
[3] Silberschatz, Galvin, Gagne - Operating Systems Concepts - John Wiley & Sons; 10 edition (2018) – ISBN 978-1-119-32091-3

Ejercicio 1. Responder:

- Usted está trabajando en CANON, quien produce cámaras de fotografía digital. En el equipo de desarrollo del sistema operativo de la cámara. Se debe decidir y desarrollar un sistema de archivos. ¿Qué estrategia de ubicación de bloques de datos para cada archivo le parece más conveniente?: ¿contiguo, posiblemente disperso con lista enlazada, o posiblemente disperso con bloques de índices?. Explique el por qué de su decisión. El sistema de archivos almacena las fotos obtenidas por la cámara.

- Un sistema UNIX tiene un sistema de archivos donde cada inodo ocupa 64 bytes. ¿Cuántos accesos a discos son necesarios para obtener la estructura i-node del archivo tareas.txt? El archivo se encuentra en:
  /export/home/alumno/so/tareas.txt
  Asuma las siguientes condiciones:
  - Los bloques en el disco son de 4KB.
  - El inodo del directorio raíz ya se encuentra en memoria (el sistema operativo ya puede acceder a él).
  - Todas las entradas de un directorio caben en un bloque del filesystem/disco.

- Un sistema de archivos de tipo-UNIX utiliza inodos que tienen 12 entradas directas a bloques, y un puntero doble-indirecto. El tamaño del bloque en disco es de 4KB y las direcciones a bloques son de 32bits. ¿Cuál es el tamaño máximo de un archivo en este sistema de archivos? Exprese el tamaño en KB y en GB.

Ejercicio 2:

Escriba un programa en C para Linux, que utilizando funciones de la biblioteca de C para realizar systems calls de acceso a archivos, lea el archivo /usr/share/doc/libsdl1.2-dev/docs.html, y presente su contenido de manera inversa. Es decir, la última letra primero, la ante última letra segunda, etc.

NO UTILICE EL TAMAÑO DE ARCHIVO DE ANTEMANO. Si se requiere conocer la cantidad de bytes que ocupa el archivo, el programa debe obtenerlo (es decir, el programa debería funcionar para cualquier archivo, no únicamente para el mencionado).

NOTA 1:
Si necesita utilizar malloc(), para reservar espacio dinámicamente, tenga en cuenta que puede luego utilizar el puntero a la memoria reservada como un arreglo. Ejemplo:

```
char *p;
int n = 10;

p = malloc(n);
if (p == NULL) {
        printf("ERROR malloc\n");
        exit(1);
}

p[9] = 'a';
printf ("la letra en la ultima posicion de p es  %c \n", p[9]);
```

NOTA 2:
Las funciones de C básicas son las que utilizó en el TP 3 ejercicio 1: open, read y close.
man 2 open
man 2 read
man 2 close

Si necesita conocer cuando se alcanzó el fin del archivo, puede consultar lo que devuelve read. Si read() devuelve 0 (cero), entonces no pudo leer más contenido. Ejemplo:

```
char c;
int r;
...etc...
i = read(fd, &c, 1);
if (i == 0)
        printf("FIN DEL ARCHIVO O UN ERROR DE OTRO TIPO\n ");
```