

Nombre:

DNI:

Segundo control de teoría

Contesta todas las preguntas justificando brevemente tu respuesta. Una respuesta sin justificar se considerará como incorrecta.

1. (2 puntos) Estructuras del sistema de ficheros

Imagina que eliminamos del sistema de ficheros la Tabla de I-nodes, y guardamos en cada entrada de la TFA el número de Inodo y el puntero a descriptor de dispositivos correspondientes. Supón que no existe ningún gestor. Contesta a las siguientes preguntas:

- a. Con esta modificación, ¿se mantiene el acceso compartido a los dispositivos lógicos?

- b. Con esta modificación, ¿se mantiene el acceso concurrente a los dispositivos lógicos?

- c. ¿Qué modificaciones se tienen que hacer en la llamada al sistema open?

- d. ¿Qué modificaciones se tienen que hacer en la llamada al sistema read?

- e. El sistema de ficheros evita que se borre un dispositivo lógico si éste está en uso. ¿Cómo se haría esta comprobación?

SOA2 (30/05/2023)

Nombre:

DNI:

2. (3 puntos) Sistema de ficheros:

Tenemos un sistema de ficheros cuyo tamaño de bloque de datos es de 1024 bytes, equivalente a 4 sectores de disco. El sistema de ficheros contiene 8192 Inodos y 4096 bloques de datos. Siempre se utilizarán Inodos para guardar la información relativa a un fichero y el número de Inodos será estático (siempre habrá 8192). Los índices a bloque ocupan 4 bytes.

En el caso de que todos los ficheros tengan exactamente 1024 bytes, indica, justificándolo, el número de ficheros máximo que podemos crear si el mecanismo de asignación de bloques a ficheros es:

a) Continuo

b) Encadenado (sin tabla)

c) Encadenado con tabla

d) Indexado

e) Indexado multinivel

f) ¿Cuántos índices caben en un bloque de índices?

SOA2 (30/05/2023)

Nombre:

DNI:

- g) ¿Hasta que tamaño de fichero la asignación encadenada sin tabla ocupará menos espacio en disco que la indexada?

- h) ¿Cuántos accesos a disco se tendrán que realizar para acceder al offset 1063004 en este sistema de ficheros si la asignación es indexada multinivel con 10 índices directos a bloques de datos, 1 indirecto y 1 doblemente indirecto dentro del Inodo?

- i) Indica, cual será el tamaño de los mapas de bits de Inodos y bloques de datos libres en esta partición.

- j) Indica cual será el tamaño de la FAT si esta partición utiliza asignación encadenada en tabla

3. (3 puntos) Syncro

Queremos implementar el código para un dispositivo que se llamará Syncro. Este dispositivo permitirá la sincronización entre threads de tal forma que cuando un thread lea este dispositivo, se quedará bloqueado hasta que alguien escriba en este dispositivo. En el caso de que haya más de un thread bloqueado, desbloqueará a uno de ellos. Tanto read como write devolverán 0.

La sucesión de llamadas a funciones que se hace hasta llegar al código del driver Syncro, para la llamada al sistema read, es:

- 1.- int sys_read(int fd, void *buffer, size_t size)
- 2.- int read_vfs(struct Inode *Inode, int offset, void *buffer, size_t size)
- 3.- int read_syncro(struct opaque* data, int offset, void *buffer, size_t size)

Por ahora supondremos que no tenemos ningun gestor.

SOA2 (30/05/2023)

Nombre:

DNI:

- a) Enumera las estructuras del sistema de ficheros a las que accede `sys_read`

- b) ¿Qué campos debería contener la estructura `struct opaque`? ¿Cómo y dónde se inicializan?

- c) Escribe el código de `read_syncro`

- d) Escribe el código de `write_syncro`

- e) ¿Cómo obtiene `read_vfs` la dirección de la función del driver `read_syncro`?

Ahora añadimos un único gestor entre el virtual file system y el driver syncro. El virtual file system no tiene ningun gestor asociado.

- f) ¿Se tiene que modificar el código de la función `read_vfs`?

- g) ¿Qué estructuras de datos se han tenido que añadir para poder enviar peticiones a este gestor?

SOA2 (30/05/2023)

Nombre:

DNI:

- h) A parte del semáforo del gestor, ¿Cuántos semáforos se han tenido que añadir para con este nuevo gestor?

- i) ¿Para qué vale el semáforo del gestor? ¿Cómo se inicializa?

- j) Escribe el código de este gestor.

- k) Dado el funcionamiento de este dispositivo, ¿sería necesario tener un gestor?

4. (2 puntos) Rendez-vous

Un Rendez-vous es un mecanismo que sirve para intentar aumentar la posibilidad de que dos partes del código se ejecuten en paralelo. Normalmente se hace de la siguiente forma para 2 threads:

```
sem_t sem1, sem2;  
sem_init(&sem1, 0);  
sem_init(&sem2, 0);
```

```
    T1  
sem_post(&sem1);  
sem_wait(&sem2);  
    código paralelo
```

```
    T2  
sem_post(&sem2);  
sem_wait(&sem1);  
    código paralelo
```

SOA2 (30/05/2023)

Nombre:

DNI:

En un mundo ideal, con un ordenador con varios procesadores, T1 y T2 con el Rendez vous, ejecutarían “código paralelo” a la vez. Pero como habéis visto durante este cuatrimestre, los sistemas operativos no tienen nada de “ideal”.

a) (0.5 puntos) ¿Por qué no se puede asegurar con un Rendez-vous que la línea “código paralelo” en los 2 threads se ejecute a la vez?

b) (0.5 puntos) ¿Se podrían hacer primero los sem_wait y después los sem_post en cada uno de los threads?

c) (1 punto) Escribe el código para hacer un Rendez-vous entre 3 threads.