SOA (17/01/2023)

Final de laboratorio

Crea un fichero que se llame “respuestas.txt” donde escribirás las respuestas para los apartados de los ejercicios del control. Indica para cada respuesta, el número de ejercicio y el numero de apartado (por ejemplo, 1.a, 1.b, ...).

Importante: para cada uno de los ejercicios tienes que partir de la versión de Zeos original que te hemos suministrado.

1. (2 puntos) Syscalls

Implementa la siguiente llamada al sistema:

```
int schedule(int pid);
```

esta llamada fuerza un cambio de contexto al proceso cuyo pid se pasa como primer parámetro. En caso de que no haya ningún proceso con el pid indicado, devolverá error. En cualquier otro caso, devolverá 0. La llamada al sistema tendrá como número de servicio el 11.

2. (2 puntos) Fork

Modifica el bucle de copia de datos de padre a hijo de la llamada al sistema sys_fork para que utilice solamente 2 páginas a partir de la dirección 0x200000.

3. (6 puntos) Call me

Queremos implementar un sistema de monitoreo de llamadas al sistema de ZeOS. Para ello, queremos implementar que, cada vez que se ejecute una llamada al sistema en un proceso, el sistema operativo ejecute una función (callback) de un proceso determinado. Para ello, se tiene que implementar una llamada al sistema, cuyo número de servicio será el 6, cuya sintaxis es:

```
int set_syscall_callback(void (*callback)(int syscall_number));
```

Esta llamada al sistema tiene un único parámetro que es la dirección del callback del proceso que la invoca, que se ejecutará cada vez que se ejecuta una llamada al sistema. Solamente puede existir una programación de este callback a la vez en el sistema. Cuando el proceso que tiene programado acaba, otro proceso puede volver a programar este callback.

La cabecera del callback es:

```
void callback(int syscall_number);
```

donde syscall_number es el número de llamada al sistema que ha forzado que se ejecute esta función.

Dentro del callback no se pueden ejecutar llamadas al sistema y, en el caso de que se haga, el comportamiento será indeterminado.

a) Indica qué estructuras de Zeos tienes que modificar para implementar esta nueva funcionalidad.
b) Indica qué nuevas estructuras de Zeos tienes que añadir.
c) Implementa el wrapper de la llamada al sistema *set_syscall_callback*.
d) Implementa *sys_set_syscall_callback*.
e) Indica cómo se tiene que modificar el handler de las llamadas al sistema para poder ejecutar código en alto nivel en el que se invocará el callback.
f) Explica los pasos que se tienen que dar para, desde modo sistema, invocar el callback definido por set_syscall_callback.
g) Indica cómo se volverá a modo sistema después de ejecutar el callback para poder seguir ejecutando la llamada al sistema.
h) Indica en qué parte de ZeOS se desprogramará el callback y como se tiene que modificar el código de sistema para conseguirlo.
i) (2 puntos) Implementa todo el mecanismo descrito.

4. (1 *punto) Generic Competences Third Language (Development Level: mid)*

The following paragraph belongs to the book *Understanding the Linux Kernel* by D. Bovet and M. Cesati:

"The translation of linear addresses is accomplished in two steps, each based on a type of translation table. The first translation table is called the Page Directory, and the second is called the Page Table.
The aim of this two-level scheme is to reduce the amount of RAM required per-process Page Tables. If a simple one-level Page Table was used, then it would require up to 2^20 entries (i.e. at 4 bytes per entry, 4MB of RAM) to represent the Page Table for each process (if the process used a full 4GB linear address space), even though a process does not use all addresses in that range. The two-level scheme reduces the memory by requiring Page Tables only for those virtual memory regions actually used by a process"

Create a text file named "generic.txt" and answer the following questions (since this competence is about text understanding, you can answer in whatever language you like):

a) What is the name of the second translation table used for translating linear addresses?
b) Is it usual that a process uses 4GB of linear address space?
c) Why a two-level linear address translation reduces the amount of memory required to store all the translations?

Entrega

Sube al Racó los ficheros "respuestas.txt" y "generic.txt" junto con el código que hayas creado en cada ejercicio.

Para entregar el código utiliza:

```
> tar zcfv examen.tar.gz ejercicio1 ejercicio2 ejercicio3 respuestas.txt generic.txt
```