SO2 (15/11/2024)

Primer control 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 número de apartado** (por ejemplo, 1.a, 1.b, ...).

Importante: para cada uno de los ejercicios tienes que partir del código suministrado de Zeos.

1. *(3 puntos)* Comprensión de Zeos

a) (1 punto) ¿En qué dirección de memoria se encuentra el PCB del 3r proceso del vector de tareas? ¿Cómo lo has hallado?

b) (1 punto) ¿Qué valor tiene el registro ESP antes de ejecutar la 1ª instrucción del código de usuario? ¿Cómo lo has hallado?

c) (1 punto) Dado el fichero *exam.s* con código en ensamblador, indica **las líneas de comandos** para compilar y linkar este fichero en la imagen de sistema.

2. *(2 puntos) Task switch*

El reputado investigador BakaBaka propone el siguiente wrapper en C para hacer el cambio de contexto:

```
1:  void task_switch(struct task_struct *new) {
2:    save_regs(new);
3:    inner_task_switch(new);
4:    restore_regs(new);
5:  }
```

Donde la función *inner_task_switch* corresponde a la vista en teoria y las funciones *save_regs* y *restore_regs* las programa en ensamblador:

```
6:  save_regs:
7:    pushl %edi
8:    pushl %esi
9:    pushl %ebx
10:   ret
11: restore_regs:
12:   popl %ebx
13:   popl %esi
14:   popl %edi
15:   ret
```

Responde a estas preguntas:

a) ¿Funcionará este código? ¿Por qué?

b) En lugar de guardar EBX, ESI y EDI en la pila nos planteamos usar un campo en el PCB. Define una estructura para guardar estos 3 registros y añádela al PCB.

c) Indica el código necesario para implementar las funciones *save_regs y restore_regs* en lenguaje C.

d) Indica cómo debe cambiar el contexto del proceso idle en su inicialización.

3. (5 puntos) Mourning my child

Queremos modificar nuestro ZeOS para permitir que la llamada a sistema exit permita devolver un código de error:

```
void exit(int error)
```

Esta llamada libera los recursos del proceso y guarda el código de error. Este código de error lo podrá consultar el padre del proceso con la llamada a sistema:

```
int waitpid(int pid, int *status);
```

que bloquea al proceso actual hasta que muera su hijo ´pid´, devolviendo el pid del proceso muerto y en ´status´ el código de error de ese proceso. Si ese hijo ya hubiera muerto, esta llamada no debe bloquearse. Si el proceso no tuviera ningún hijo con ese pid esta llamada debe devolver error. Para implementar esta llamada a sistema debe usarse la interrupción 130 (en lugar de los mecanismos ya implementados) y debe ejecutar la rutina de servicio directamente pues será el único servicio accesible mediante esta interrupción.
La llamada exit debe notificar a sus hijos que su padre ha muerto y delegar el parentesco al proceso idle que los eliminará cuando mueran.

Se pide:

a) (0,5 puntos) Modifica el wrapper de la llamada *exit*.
b) (0,5 puntos) Implementa el código del wrapper de la llamada *waitpid.*
c) (0,5 puntos) Implementa el código del handler de esta llamada a sistema.
d) (0,5 puntos) Indica qué estructuras de datos se tienen que añadir. Indica también las estructuras que deben ser modificadas. Añade el código necesario para inicializarlas.
e) (0,5 puntos) Implementa la rutina *void block(void)* para bloquear al proceso actual .
f) (0,5 puntos) Implementa la rutina *void unblock(struct task_struct*pcb)* para desbloquear un proceso pasado como parámetro.
g) (0,5 puntos) Modifica el código de la rutina *sys_exit*.
h) (1 punto) Implementa el código de la rutina de servicio *sys_waitpid.*
i) (0,5 puntos) ¿Es necesario modificar alguna otra llamada a sistema o parte del sistema para implementar por completo esta funcionalidad? Si es así implementa los cambios necesarios.

Puedes suponer que tienes la función *struct task_struct* find_task_by_pid(int pid)* que retorna el puntero al PCB del proceso con pid PID o NULL si no lo encuentra.

4. Entrega

Sube al Racó los ficheros “respuestas.txt” junto con el código que hayas creado en cada ejercicio.

Para entregar el código utiliza:

```
> tar zcfv examen.tar.gz zeos
```