Segundo Parcial

Primer Cuatrimestre 2022

Normas generales

- El parcial es INDIVIDUAL
- Puede disponer de la bibliografía de la materia y acceder al repositorio de código del taller de system programming, desarrollado durante la cursada
- Las resoluciones que incluyan código, pueden usar assembly o C. No es necesario que el código compile correctamente, pero debe tener un nivel de detalle adecuado para lo pedido por el ejercicio.
- Numere las hojas entregadas. Complete en la primera hoja la cantidad de hojas entregadas
- Entregue esta hoja junto al examen. La misma no se incluye en el total de hojas entregadas.
- Luego de la entrega habrá una instancia coloquial de defensa del examen

Enunciado - Tema 1

Ejercicio 1

Estructura general del sistema

En un sistema similar al que implementamos en los talleres del curso (modo protegido con paginación activada), se necesita implementar una llamada al sistema que permita modificar la próxima tarea a ejecutar de la forma solicitada.

La próxima tarea a ejecutar es aquella tarea, distinta a la actual, retornada por la llamada a `sched_next_task` en la rutina de interrupción del reloj.

Se busca ampliar el sistema para que una tarea pueda modificar el valor del registro `edx` de la tarea próxima. Para ello se define una syscall que al ser llamada almacenará el valor a ser escrito **a la hora de realizar el cambio de tarea** (la solicitud y la modificación ocurren en diferido). Dicho en otras palabras, cuando la tarea próxima retome la ejecución **a nivel de usuario** su `edx` cambiará sólo si la tarea anterior utilizó esta llamada al sistema. El valor que tomará el registro `edx` se pasará por parámetro a la syscall.

1. Modificar las estructuras del sistema necesarias para que todas las tareas puedan invocar a esta nueva syscall, indique las modificaciones que introduciría, y en qué archivos del sistema implementado en los talleres las harían.
2. Implementar la rutina correspondiente a dicha syscall y definir variables externas si fuese necesario.
3. Modificar la rutina de interrupción de reloj para que si se llamó a la syscall en la tarea actual, se haga la modificación correspondiente sobre el registro de la tarea próxima.

Ejercicio 2

En un sistema con características similares al del taller, diseñe una función que tome una dirección virtual y la dirección de un *page directory table* y devuelva la dirección física correspondiente. Además, debe devolver en `attrs` un entero que contenga en sus 3 bits menos significativos el estado **combinado** de los flags U/S, R/W y P, respectivamente, en función de los atributos del PDE y PTE correspondientes. La firma de la función en cuestión es:

```
uint32_t getPhysical(uint32_t virtual, pd_entry_t* pdt, uint32_t *attrs)
```

Estando declarado el tipo `pd_entry_t`, como en el taller. Se repite su declaración aquí:

```
typedef struct pd_entry_t {
        uint32_t attrs : 12;
        uint32_t pt : 20;
} __attribute__((packed)) pd_entry_t;
```

En caso de utilizar tipos de datos, macros, definiciones y/o funciones provistas en el taller, enumere sus declaraciones y fundamente su uso.

Ejercicio 3

¿Porqué un sistema que utilice dos o más niveles de privilegio diferentes debe usar una TSS aun cuando la conmutación de tareas es manual?