Sistemas Operativos

Primer cuatrimestre 2019

Trabajo Práctico Nro. 3 (obligatorio): Estructuras de administración de recursos

Introducción

En el TP Nº 2 se implementó un kernel con procesos, IPCs, Memory Management y Scheduling. Ahora deberán implementar estructuras para agregarle mejor administración de recursos a nuestro sistema.

Grupos

Se realizará con los grupos ya establecidos.

Requisitos Previos

TP Nº 2 completado.

Requerimientos

Deberán implementar los requerimientos listados a continuación:

- Pipes
- Mejoras al Physical Memory Management
- Mejoras al scheduler
- Filósofos comensales

Pipes

El kernel debe ofrecer un sistema de IPC por pipes, implementando una solución del problema del productor y el consumidor, asimismo la shell debe poder conectar dos procesos de forma similar a la de Unix, por ejemplo ./producer | ./consumer.

Syscalls involucradas

- Crear, Escribir, Leer y Cerrar pipes.

Mejoras al Physical Memory Management

Implementar Buddy Allocation para poder reservar varias páginas físicas contiguas.

Syscalls involucradas

- Las mismas que en el TP anterior

Mejoras al scheduler

Deberán implementar un scheduler que soporte manejo de prioridades. Las mismas pueden ser modificadas por cualquier proceso utilizando una syscall. También deberán crear un programa de usuario que funcione como un wrapper de esta syscall, tal como en unix con el comando de usuario **nice** y la syscall **nice**. El scheduler deberá evitar inanición y deberá tener presente los locks adquiridos por cada proceso para evitar inanición por inversión de prioridades.

Syscalls involucradas

- Modificar prioridad de proceso

Filósofos comensales

Deberán proveer una aplicación de usuario que implemente una solución al problema de los filósofos comensales y que permita agregar y eliminar comensales en runtime maximizando la concurrencia, es decir, que no se tengan que bloquear todos los filósofos para agregar o eliminar.

Aplicaciones de Userspace

Para mostrar el cumplimiento de todos los requisitos anteriores, deberán desarrollar varias aplicaciones, que **muestren el funcionamiento del sistema** llamando a las distintas system calls. Por ejemplo, para mostrar la protección de memoria, debería haber una aplicación que intente acceder a una zona inválida, y sea bloqueada por el kernel.

Consideraciones

Es necesario que programen de forma modular y desacoplada. Asimismo se deberá respetar la separación a nivel binario y memoria entre kernelspace y userspace, las aplicaciones de usuario se comunican entre sí y con el kernel solo a través de system calls.

Armar casos de prueba, transformarlos en testeos unitarios y derivar las features del sistema operativo no es mandatorio, pero si es muy recomendado, en especial para los algoritmos más complejos.

Asimismo, si se programa de forma desacoplada y a partir de testeos unitarios, utilizar valgrind para revisar errores de acceso de memoria se vuelve trivial, esto es importante ya que debuggear un sistema operativo en runtime es posible, pero muy trabajoso, debería ser la última opción.

Informe

Se deberá presentar un informe en formato pdf (NO entregarlo impreso), en el cual se desarrollen de forma breve, los siguientes puntos:

- Decisiones tomadas durante el desarrollo, por ejemplo, qué algoritmo de scheduling eligieron, cómo implementaron el page alocator, etc.
- Instrucciones de compilación y ejecución.
- Limitaciones.
- Problemas encontrados durante el desarrollo y cómo se solucionaron.
- Citas de fragmentos de código reutilizados de otras fuentes.

Entorno de compilación

Es un requisito obligatorio para la compilación, utilizar la imagen provista por la cátedra:

```
docker pull agodio/itba-so:1.0
```

Evaluación

La evaluación incluye y no se limita a los siguientes puntos:

- [1] Deadline.
- [4] Funcionalidad (Mandatorio).
- [3] Calidad de código.
- [1] Tests.
- [1] Informe.
- Defensa.

Entrega

Fecha: 03/06/2019 hasta las 23:59.
Se habilitará la actividad “TP3” en el campus donde se podrá subir el material requerido. En caso de tener algún problema con la entrega en Campus, enviarlo por mail a los docentes a cargo de la clase práctica.
Entregables: Link del repositorio especificando el hash del commit y la rama correspondiente a la entrega y el informe.
Defensa del trabajo práctico: Grupal, obligatoria y con nota individual 10/06/2019.