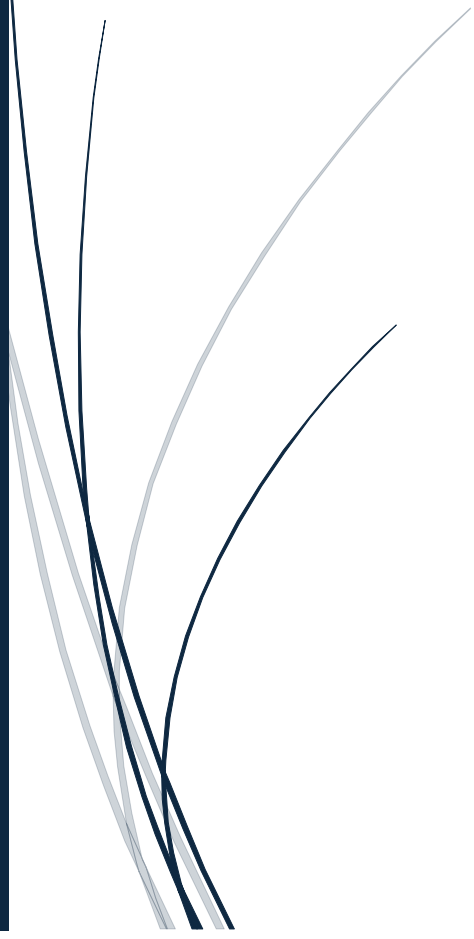
27-4-2024

Procesos POSIX

Práctica 6 – Sistemas Distribuidos



Julián Blanco González
UNIVERSIDAD DE EXTREMADURA

Índice

Introducción 1
Código 2
Librerías 2
Variables 2
Obtener el valor de la prioridad mínima y máxima de cada política (punto 1) 3
Quantum de tiempo de Round Robin (punto 1) 3
Política de planificación y prioridad del proceso actual (punto 2) 3
Cambiar la política y la política de proceso actual (punto 3) 4
Cambiar la política a RR y asignar la prioridad máxima (punto 4) 4
Compilar y Ejecución por pantalla 4
Bibliografía 5

Introducción

Implementar un programa en c que obtenga información sobre los valores de política y prioridades del proceso en curso.

• Visualizar que valor tiene cada política de planificación (SCHED_OTRHER, SCHED_FIFO, SCHED_RR). Para cada política, visualizar el valor de prioridad mínimo, máximo y en el caso de Round Robin, el valor del “quantum” de tiempo.

• Mostrar la política de planificación y la prioridad asignada al proceso actual.

• Según los valores mostrados, cambiar la política y la prioridad del proceso actual. A continuación, visualizar los valores.

• Modificar la política a SCHED_RR y asignar la prioridad máxima. A continuación, mostrar que se hayan realizado los cambios correctamente.

Código

A continuación, voy a explicando por partes, el código que he creado para poder hacer esta práctica.

Librerías

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sched.h>
```

La novedad es la biblioteca “sched.h” que permite controlar y gestionar la planificación de procesos e hilos del sistema operativo.

Variables

```
int min_priority, max_priority, current_policy, current_priority;
struct sched_param param;
struct timespec quantum;
```

min_priority: Almacena la prioridad mínima disponible para una política de planificación específica.

max_priority: Almacena la prioridad máxima disponible para una política de planificación específica.

current_policy: Almacena la política de planificación actual del proceso.

current_priority: Almacena prioridad actual del proceso.

param: Estructura sched_param que se utiliza para almacenar y especificar los parámetros de planificación, como la prioridad.

quantum: Estructura timespec que se utiliza para almacenar el intervalo de tiempo del "quantum" para la política de planificación Round Robin (SCHED_RR).

Obtener el valor de la prioridad mínima y máxima de cada política (punto 1)

```
min_priority = sched_get_priority_min(SCHED_OTHER);
max_priority = sched_get_priority_max(SCHED_OTHER);
printf("Prioridades para SCHED_OTHER: mínima = %d, máxima = %d\n\n", min_priority, max_priority);

min_priority = sched_get_priority_min(SCHED_FIFO);
max_priority = sched_get_priority_max(SCHED_FIFO);
printf("Prioridades para SCHED_FIFO: mínima = %d, máxima = %d\n\n", min_priority, max_priority);

min_priority = sched_get_priority_min(SCHED_RR);
max_priority = sched_get_priority_max(SCHED_RR);
printf("Prioridades para SCHED_RR: mínima = %d, máxima = %d\n\n", min_priority, max_priority);
```

Con la función “sched_get_priority_min” y “sched_get_priority_max” pasándole por parámetro la política de la cual quiero el valor, obtengo los valores mínimos y máximos de cada una de las 3 políticas.

Quantum de tiempo de Round Robin (punto 1)

```
if (sched_rr_get_interval(0, &quantum) != 0) {
    perror("Error obteniendo el intervalo de tiempo del \"quantum\" para SCHED_RR");
    exit(EXIT_FAILURE);
}

printf("Intervalo de tiempo del \"quantum\" para SCHED_RR: %ld segundos, %ld nanosegundos\n\n",
        quantum.tv_sec, quantum.tv_nsec);
```

Con la función de “sched_rr_get_interval” con el parámetro 0 (proceso actual) y &quantum (puntero a estructura para obtener el tiempo), obtengo el intervalo de tiempo. Si da error, muestro un mensaje de error y si no, muestro el tiempo en segundos y nanosegundos.

Política de planificación y prioridad del proceso actual (punto 2)

```
current_policy = sched_getscheduler(0);
if (current_policy == -1) {
    perror("Error obteniendo la política actual");
    exit(EXIT_FAILURE);
}

if (sched_getparam(0, &param) != 0) {
    perror("Error obteniendo la prioridad actual");
    exit(EXIT_FAILURE);
}

printf("Política actual: %d, Prioridad actual: %d\n\n", current_policy, param.sched_priority);
```

Obtengo la política actual con la función “sched_getscheduler” con el parámetro 0 (proceso actual).

Obtengo con la función “sched_getparam” con parámetros 0 (proceso actual) y &param (puntero a estructura) la prioridad del proceso actual.

Cambiar la política y la política de proceso actual (punto 3)

```c
// Cambiar la política y la prioridad del proceso actual
param.sched_priority = min_priority;
if(sched_setscheduler(0, SCHED_FIFO, &param) == -1) {
    perror("sched_setscheduler");
    exit(EXIT_FAILURE);
}

// Mostrar los valores después del cambio
current_policy = sched_getscheduler(0);
sched_getparam(0, &param);
current_priority = param.sched_priority;
printf("Después del cambio - Política de planificación: %d, Prioridad: %d\n", current_policy, current_priority);
```

La prioridad la cambio a la prioridad mínima y la planificación FIFO la cambio usando la función “sched_setscheduler” con los parámetros 0 (proceso actual), SCHED_FIFO (política a cambiar) y &param (puntero a estructura). Luego recupero con ambas funciones get, y muestro los valores.

Cambiar la política a RR y asignar la prioridad máxima (punto 4)

```c
// Modificar la política a SCHED_RR y asignar la prioridad máxima
param.sched_priority = max_priority;
if(sched_setscheduler(0, SCHED_RR, &param) == -1) {
    perror("sched_setscheduler");
    exit(EXIT_FAILURE);
}

// Mostrar los valores después del cambio a SCHED_RR
current_policy = sched_getscheduler(0);
sched_getparam(0, &param);
current_priority = param.sched_priority;
printf("Después del cambio a SCHED_RR - Política de planificación: %d, Prioridad: %d\n", current_policy, current_priority);
```

Cambio la prioridad a la máxima y con la función “sched_setscheduler”, pongo la política Round Robin. Y por último, muestro los valores de la política y la prioridad.

Compilar y Ejecución por pantalla

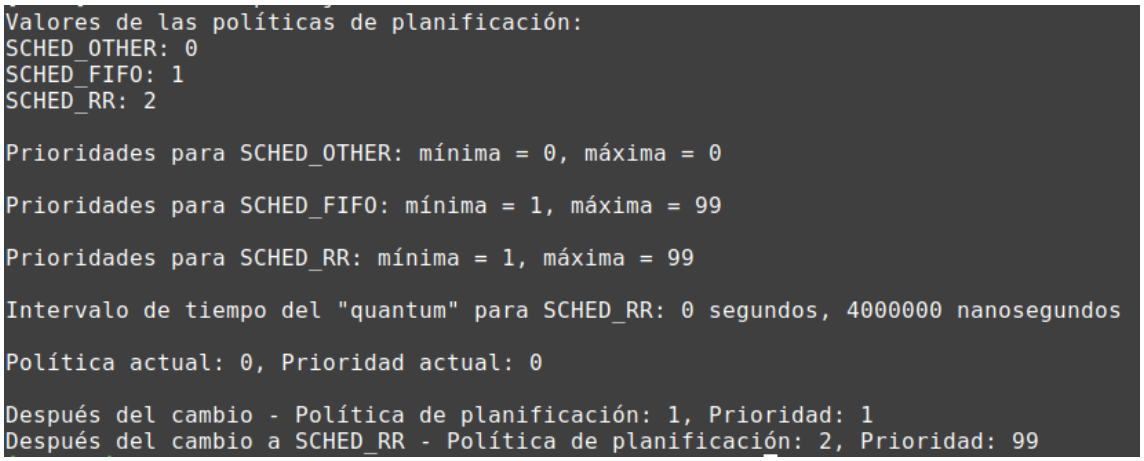
Para compilar el archivo, hay que usar el siguiente comando:

```
julian@julian-VirtualBox:~/Escritorio/Distribuidos/S6$ gcc -o seis 6_2.c
```

Una vez creado el ejecutable, hay que ejecutarlo usando los privilegios de superusuario (sudo) para poder cambiar la política y la prioridad de un proceso:

```
julian@julian-VirtualBox:~/Escritorio/Distribuidos/S6$ sudo ./seis
```

Una vez puesta la contraseña, la ejecución por pantalla es esta:



Punto 1: mostrar los valores de la política de planificación, sus prioridades mínimas y máximas y el intervalo del tiempo de quantum para la política Round Robin

Punto 2: mostrar la política de planificación actual y la prioridad del proceso actual

Punto 3: Cambiar la política y la prioridad y mostrarla. En mi caso, la he cambiado a SCHED_FIFO, que tiene como valor de política de planificación 1 (se puede ver arriba) y le he asignado el valor mínimo de prioridad (1).

Punto 4: cambiar a la política de Round Robin con la prioridad máxima. Se puede ver que la política de Round Robin es la 2, y la prioridad máxima es 99.

Bibliografía

Realizar la práctica

PDF de la práctica: Procesos POSIX.pdf

Código asociado: ejemplo1.c

Ver la estructura de la política de Round Robin

https://manual.cs50.io/2/sched_rr_get_interval