SISTEMAS OPERATIVOS

Tecnológico de Monterrey, Campus Querétaro

Laboratorio 5 - Módulo Kernel de Linux para listado de procesos

1. Introducción

En este laboratorio, escribirás un módulo kernel que despliega todas las tareas actuales en un sistema Linux. Asegúrese de revisar el laboratorio 4 (Módulos de Kernel), que trata sobre la creación de módulos del kernel de Linux, antes de comenzar este laboratorio. El proyecto se puede completar utilizando la máquina virtual Linux proporcionada en el curso.

2. Iterando sobre procesos de forma lineal.

Como se muestra en la Sección 3.1 del libro, el PCB en Linux está representada por la estructura `task_struct`, que se encuentra en el archivo cabecera `<linux/sched.h>`. En Linux, la macro `for each process()` permite una fácil iteración sobre todas los procesos actuales en el sistema:

```
#include <linux/sched.h>
struct task struct *task;

for each process(task) {
  /* on each iteration task points to the next task */
}
```

Los diversos campos de la estructura `task_struct` se pueden mostrar a medida que el programa recorre la lista de procesos empleando la macro `for each process()`.

2.1. Ejercicio a realizar

Diseña un módulo kernel que itere a través de todas los procesos que hay en el sistema usando la macro `for each process()`. En particular, muestra el nombre del proceso (conocido como nombre ejecutable), el estado y la identificación del mismo. (Probablemente tendrás que leer la estructura `task_struct` en `<linux/sched.h>` para obtener los nombres de estos campos). Escribe este código en el punto de entrada del módulo para que su contenido aparezca en el búfer de registro del núcleo, que se puede ver usando el comando dmesg. Para verificar que tu código está funcionando correctamente, compara el contenido del buffer del registro del kernel con el resultado del siguiente comando que despliega todas los procesos que hay en el sistema:

```
ps -el
```

Los dos desplegados deberían ser muy similares. Sin embargo, dado que los procesos son dinámicos, es posible que algunos procesos aparezcan en una lista, pero no en la otra.

3. Iterando sobre procesos usando búsqueda en profundidad

La segunda parte de este laboratorio implica iterar sobre todos los procesos que hay en el sistema usando una búsqueda profunda sobre el árbol de procesos.

Linux mantiene su árbol de procesos como una serie de listas. Si examinas la estructura `task_struct` en `<linux/sched.h>`, verás dos campos del tipo `list_head`: `child` and `sibling`. Estos campos son apuntadores a una lista de hijos del proceso, así como a sus hermanos. Linux también mantiene referencias a `init task` (`struct task_struct init_task`). Usando esta información, así como las macros sobre listas, podemos iterar sobre los elementos secundarios de `init` de la siguiente manera:

```
struct task struct *task;
struct list head *list;
list for each(list, &init task->children) {
  task = list entry(list, struct task struct, sibling);
  /* task points to the next child in the list */
}
```

La macro `list_for_each` recibe dos parámetros, ambos del tipo `struct list_head`:

- Un apuntador a la lista que va a iterar.
- Un apuntador al nodo inicial de la lista que se va a iterar.

En cada iteración de `list_for_each`, el primer parámetro se direcciona a la estructura `list` del siguiente hijo. Luego usamos este valor para obtener cada estructura almacenada en la lista usando la macro `list_entry()`.

3.1. Ejercicio a realizar

A partir del proceso `init`, diseña un módulo kernel que itere sobre todos los procesos en el sistema usando un búsqueda en profundidad. Al igual que en la primera parte de este laboratorio, muestra el nombre, el estado y el pid de cada proceso. Realiza esta iteración en el módulo de entrada del kernel para que el resultado aparezca en el búfer de registro del kernel.

Si tu salida despliega todos los procesos en el sistema, podrás ver muchos más procesos que los que aparecen con el comando `ps -ael`. Esto es porque algunos hilos aparecen como hijos pero no se muestran como procesos ordinarios. Por lo tanto, para verificar tu salida, usa el comando:

```
ps -eLf
```

Es comando despliega todos los procesos, incluidos los hilos, que hay en el sistema. Para verificar que efectivamente has realizado una iteración apropiada, tendrás que examinar las relaciones entre los diversos procesos generados por el comando `ps`.