Universidad del Valle de Guatemala
Sistemas Operativos
Eduardo Ramírez 19946
Semestre I, 2022

LABORATORIO #5

- **Funcionamiento y sintaxis de uso de structs.**

    1. Esta relacionado a POO y sirve como objeto

    2. Esta definido de la siguiente manera:

```
struct nombre
{
        char
        c;int i;
}
```

    3. Es una coleccion de variables que se relacionan bajo un mismo nombre, y pueden tener distintas variables con distintos tipos de datos
    4. El funcionamiento es similar al de una interfaz de java.

- **Propósito y directivas del preprocesador.**

    - El pre procesador esta definido por cpp y es lo primero que se ejecuta en el compilador para traer las directivas de:

        - #define label text
        - #elif condition
        - #else
        - #endif
        - #error "message"
        - #if condition
        - #ifdef label
        - #ifndef label
        - #include {"filename" | <filename>}
        - #message "message"
        - #undef label

- **Diferencia entre *y &en el manejo de referencias a memoria (punteros).**

    - Utilizamos el * para obtener el valor al cual apunta un puntero
    - Utilizamos el & para obtener la dirección de memoria de cualquier variable

- **Propósito y modo de uso de APT y dpkg.**

    - Apt es una librería que contiene todos los paquetes del sistema, se le pueden instalar o actualizar paquetes con el comando apt-get update o eliminar paquetes.
    - Dpkg es el manejador de paquetes de debian con el cual podemos eliminar, instalar o descargar paquetes para debian y Linux.

- **¿Cuál es el propósito de los archivos sched.hmodificados?**

    - Esto crea las distintas clases de procesos que pueden existir, para esto se crean macros que identifican las políticas de calendarización que se van a usar.

- **¿Qué es una *task* en Linux?**

    - Una tarea o task en Linux es todo aquello que interactúa con el CPU y el task entity y esto se define como un task programado porque se encarga de programar la¿os tasks de kernel.

- **¿Cuál es el propósito de task_structy cuál es su análogo en Windows?**

    - Este es una tarea dentro de otras que ya están siendo realizadas o ejecutadas y estas guardan toda la información que contiene un proceso, en Windows es conocido como el EPROCESS y esta también contiene un KPROCESS la cual tiene la información del kernel.

- **¿Qué información contiene sched_param?**

    - Sched_param contiene la estructuración con los parámetros útiles para la calendarización.

- **¿Para qué sirve la función rt_policyy para qué sirve la llamada unlikelyen ella?**

    - Este determina la política que tiene el calendarizador y al momento de llamar unlikely hace que el procesador de instrucciones que hagan que una condicional no funcione correctamente.

- **¿Qué tipo de tareas calendariza la política EDF, en vista del método modificado?**

    - Si la política establecida hace que ocurra algún evento esta buscara dentro de la cola de procesos al que tenga la fecha límite más reciente para priorizarlo.

- **Explique el contenido de la estructura casio_task.**

    - Esto contiene una etiqueta absolute_deadline
    - Dos nodos que se iteran
    - Tasks a realizar

- **Explique el propósito y contenido de la estructura casio_rq.**

    - Esto coordina los procesos según su prioridad y la forma de realizar esto es por medio de una lista que debe ser recorrida de manera ascendente y ordenada constantemente para que los procesos prioritarios sean los primeros en realizarse.

- **¿Qué es y para qué sirve el tipo atomic_t? Describa brevemente los conceptos de operacionesRMW (*read-modify-write*) y *mappeo* de dispositivos en memoria (MMIO).**

    - RMW, es un operador que lee la location en memoria y escribe valores en forma simultánea, esta también evita las race conditions.
    - MMIO es un direccionamiento de memoria y este permite construir el sistema de memoria que usa el cpu para que este use los registros del chip de memoria como su fueran las posiciones del la ram
    - Atomic_t, una variable int que cuenta cuantas operaciones se garantizan ser atómicas y que no exista un bloqueo.

- **¿Qué indica el campo .nextde esta estructura?**

    - Esto nos indica la dirección de memoria de la variable, la cual organiza los módulos que planifican la prioridad de la lista de procesos.

- **¿Por qué se guardan las casio_tasksen un *red-black tree* y en una lista encadenada?**

    - Estas realizan llamadas cada vez que una casio_task es ejecutada y esto va de la mano con find_casio la cual nos indica el puntero a struct_casio que se vincula en la lista. Luego se actualiza la deadline para luego insertar en el red-black tree

- **¿Cuándo *preemptea* una casio_taska la *task* actualmente en ejecución?**
    - Esta selecciona el task que se ejecutara dentro de un proceso.

- **¿Qué información contiene el archivo system que se especifica como argumento en la ejecución de casio_system?**

    - El tiempo de ejecución de cada tarea o las tareas en general.

- **Investigue el concepto de aislamiento temporal en relación con procesos. Explique cómo el calendarizador SCHED_DEADLINE, introducido en la versión 3.14 del *kernel* de Linux, añade al algoritmo EDF para lograr aislamiento temporal.**

    - SCHED_DEADLINE es un algoritmo Edit Deadline First el cual reserv los recursos y declara las tareas como independientes a sus requisitos en tiempo. El kernel acepta estas tareas en el calendarizado después hayan pasado una prueba de calendarizadores.

    - Es un mecanismo de aislamiento de los procesos, este mecanismo se ejecuta de forma separada. Se utiliza para ejecutar código nuevo o software que pueda contener archivos que dañen el computador, de esta manera se permite controlar los recursos proporcionados a los programas huésped que se están ejecutando en un espacio temporal de memoria.