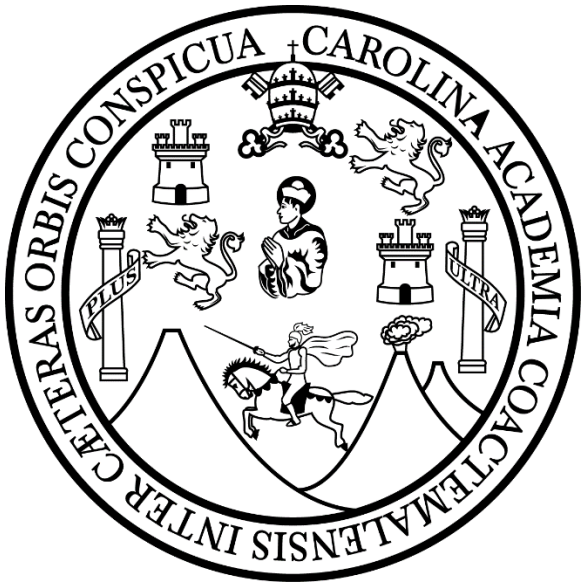
Universidad de San Carlos de Guatemala
Facultad de Ingeniería.
Sistemas Operativos 1
Sección “N”
Ing. Sergio Méndez.
Auxiliar. Bryan Ordoñez




# MODULOS KERNEL, PLANIFICACIÓN DE PROCESOS Y MONITOREO DE RECURSO

201314474 Bayron Romeo Axpuac Yoc
201314160 Luis Manuel Morales López.

Guatemala Junio 2020

# LINUX

Linux es un sistema operativo de software libre (no es propiedad de ninguna persona o empresa), por ende no es necesario comprar una licencia para instalarlo y utilizarlo en un equipo informático. Es un sistema multitarea, multiusuario, compatible con UNIX, y proporciona una interfaz de comandos y una interfaz gráfica, que lo convierte en un sistema muy atractivo y con estupendas perspectivas de futuro. Al ser software libre, el código fuente es accesible para que cualquier usuario pueda estudiarlo y modificarlo. La licencia de Linux no restringe el derecho de venta, por lo que diversas empresas de software comercial distribuyen versiones de Linux. Además de esto, este sistema cuenta con muchas distribuciones y gestores de ventanas para el entorno gráfico.

El sistema operativo Linux fue desarrollado por Linus Torvalds, y se basa en el sistema Minix que a su vez está basado en el sistema Unix, Torvalds fue añadiéndole herramientas y utilidades, haciéndolo operativo. A partir de la primera versión de Linux el sistema ha sido modificado por miles de programadores de todo el mundo, bajo la coordinación de su creador. El nombre de Linux proviene del nombre de su autor Linus y del sistema operativo UNIX. No obstante, su verdadero nombre es GNU/Linux, ya que el sistema se distribuye bajo licencia GNU GPL (General Public License).

La estructura del Linux está basada en un micronúcleo híbrido que ejecuta los servicios más básicos del sistema operativo. **El Kernel** es el núcleo del sistema; la parte que interactúa directamente con el hardware, administrando todos los recursos de éste, como la memoria, el microprocesador, los periféricos, etc. El Kernel de Linux, es el corazón del sistema operativo. Sin el Kernel, sencillamente no podríamos realizar ninguna tarea, ya que se encarga principalmente de que el software y el Hardware de nuestro PC funcionen correctamente y puedan interactuar entre ellos. Las principales funciones del Kernel, son las siguientes:

- Administrar la memoria RAM, para que puedan funcionar todos los programas y procesos en ejecución.
- Administrar el tiempo de procesador, que es utilizado por los procesos en ejecución.
- Gestionar el acceso y uso de los diferentes periféricos conectados al ordenador.

Una de las ventajas que tiene el kernel de Linux es que es posible actualizarlo sin que afecte al resto del sistema operativo, con un par de comandos (usando el usuario root) en el Terminal estaríamos logrando esto en un par de minutos o incluso más sencillo mediante el Centro de Software, aunque esto depende de la distribución que elijamos. Actualizando únicamente el kernel del sistema, tendríamos no solo un equipo más estable, también más seguro y más rápido, todo ello en varios minutos.

# EL SISTEMA DE ARCHIVOS PROC

El sistema de archivos /proc contiene un sistema de archivos imaginario o virtual. Este no existe físicamente en disco, sino que el núcleo lo crea en memoria. Se utiliza para ofrecer información relacionada con el sistema (originalmente acerca de procesos, de aquí su nombre). Algunos de los archivos más importantes se encuentran explicados a continuación. La mayoría de los archivos en el sistema de archivos proc son de solo lectura, pero algunos los archivos se pueden escribir, lo que permite cambiar las variables del kernel. El kernel de Linux tiens dos funciones primarias: controlar el acceso a los dispositivos físicos del ordenador y establecer cuándo y cómo los procesos interactuarán con estos dispositivos. El directorio /proc/ también llamado el sistema de archivos proc, contiene una jerarquía de archivos especiales que representan el estado actual del kerne, permitiendo a las aplicaciones y usuarios mirar detenidamente en la vista del kernel del sistema. Dentro del directorio /proc/, se puede encontrar una gran cantidad de información con detalles sobre el hardware del sistema y cualquier proceso que se esté ejecutando actualmente. Además, algunos de los archivos dentro del árbol de directorios /proc/ pueden ser manipulados por los usuarios y aplicaciones para comunicar al kernel cambios en la configuración.

- /proc/1: Un directorio con información acerca del proceso número 1. Cada proceso tiene un directorio debajo de /proc cuyo nombre es el número de identificación del proceso (PID).
- /proc/cpuinfo: Información acerca del procesador: su tipo, marca, modelo, rendimiento, etc.
- /proc/devices: Lista de controladores de dispositivos configurados dentro del núcleo que está en ejecución.
- /proc/dma: Muestra los canales DMA que están siendo utilizados.
- /proc/filesystems: Lista los sistemas de archivos que están soportados por el kernel.
- /proc/interrupts: Muestra la interrupciones que están siendo utilizadas, y cuantas de cada tipo ha habido.
- /proc/ioports: Información de los puertos de E/S que se estén utilizando en cada momento.
- /proc/kcore: Es una imagen de la memoria física del sistema. Este archivo tiene exactamente el mismo tamaño que la memoria física, pero no existe en memoria como el resto de los archivos bajo /proc, sino que se genera en el momento en que un programa accede a este. (Recuerde: a menos que copie este archivo en otro lugar, nada bajo /proc usa espacio en disco).
- /proc/kmsg: Salida de los mensajes emitidos por el kernel. Estos también son redirigidos hacia syslog.
- /proc/ksyms: Tabla de símbolos para el kernel.
- /proc/loadavg: El nivel medio de carga del sistema; tres indicadores significativos sobre la carga de trabajo del sistema en cada momento.

- /proc/meminfo: Información acerca de la utilización de la memoria física y del archivo de intercambio.
- /proc/modules: Indica los módulos del núcleo que han sido cargados hasta el momento.
- /proc/net: Información acerca del estado de los protocolos de red.
- /proc/self: Un enlace simbólico al directorio de proceso del programa que esté observando a /proc. Cuando dos procesos observan a /proc, obtienen diferentes enlaces. Esto es principalmente una conveniencia para que sea fácil para los programas acceder a su directorio de procesos.
- /proc/stat: Varias estadísticas acerca del sistema, tales como el número de fallos de página que han tenido lugar desde el arranque del sistema.
- /proc/uptime: Indica el tiempo en segundos que el sistema lleva funcionando.
- /proc/versión: Indica la versión del núcleo

Conviene aclarar que aunque los archivos anteriores tienden a ser archivos de texto fáciles de leer, algunas veces pueden tener un formato que no sea fácil de interpretar. Por ello existen muchos comandos que solamente leen los archivos anteriores y les dan un formato distinto para que la información sea fácil de entender. Por ejemplo, el comando free, lee el archivo /proc/meminfo y convierte las cantidades dadas en bytes a kilobytes (además de agregar un poco más de información extra).

# MÓDULOS DEL KERNEL

El kernel de Linux tiene un diseño modular. En el momento de arranque, sólo se carga un kernel residente mínimo en memoria. Por ello, cuando un usuario solicita alguna característica que no está presente en el kernel residente, se carga dinámicamente en memoria un módulo kernel, también conocido algunas veces como un controlador. Durante la instalación, se prueba el hardware en el sistema. Basado en esta prueba y en la información proporcionada por el usuario, el programa de instalación decide qué módulos necesita cargar en el momento de arranque. El programa de instalación configura el mecanismo de carga dinámica para que funcione de forma transparente. En un sistema Linux la interacción final con dispositivos la realizan los controladores o el kernel. Dicho de otra forma, un dispositivo sólo podrá ser usado si el kernel lo soporta o si existe un controlador capaz de controlarlo y si se configura apropiadamente para hacerlo. Las fuentes en C de cada versión del kernel cuentan con controladores para diversos dispositivos. Cuando se compila una versión, algunos de esos controladores pueden unirse con el kernel mismo (estáticamente), otros pueden dejarse como módulos para cargarse/descargarse cuando la parte estática del kernel este operando, otros pueden ser excluidos del proceso de compilación (y por lo tanto no podrán ser usados ni cuando el kernel esté operando).

# MÓDULOS DEL KERNEL (Proyecto)

- **Módulo de Memoria(sysinfo):** Este módulo deberá sobrescribir un archivo en el directorio /proc con la siguiente información:
    - Carné: Numero carnet del Estudiante
    - Nombre: Nombre del estudiante
    - Memoria total Memoria total utilizada en el momento en MB
    - Memoria libre Memoria total libre en MB % de memoria utilizada Porcentaje de utilización de memoria.

  Descripción: Este módulo se realizó gracias a la ayuda de las diferentes librerías y/o funciones.
    - sysinfo () devuelve ciertas estadísticas sobre el uso de memoria e intercambio, como así como el promedio de carga, con esta herramienta su creo un elemento tipo struct, a través de este estruct y operaciones matemáticas podemos obtener el total de la memoria utilizada y la memoria libre, por medio del atributo totalram y freeram.
    - La función printk está definida en el kernel de Linux y está disponible para los módulos. Es equivalente a la función printf con una diferencia menor que printk no admite tener soporte de punto flotante.
    - seq_printf (), que funciona de manera similar a printk (), pero esto requiere el puntero seq_file como argumento. Si seq_printf devuelve un valor distinto de cero, significa que el búfer se ha llenado y la salida se está descartando. Gracias a esta función logramos escribir un archivo dentro de la carpeta proc.
- **Módulo CPU(task_struct):** Este módulo deberá sobrescribir un archivo en el directorio /proc con la siguiente información de encabezado:
    - Carné Numero de carné del estudiante
    - Nombre Nombre del estudiante

  Posterior a esto deberá listar todos los procesos, mostrando:
    - PID: Identificador del proceso
    - Nombre: Nombre del proceso
    - Estado: Estado en el que se encuentra el proceso
    - Hijos: Lista de procesos hijos

  Descripción: Este módulo se realizó gracias a la ayuda de las diferentes librerías y/o funciones.
    - La función printk está definida en el kernel de Linux y está disponible para los módulos. Es equivalente a la función printf con una diferencia menor que printk no admite tener soporte de punto flotante.
    - seq_printf (), que funciona de manera similar a printk (), pero esto requiere el puntero seq_file como argumento. Si seq_printf devuelve un valor distinto de cero, significa que el búfer se ha llenado y la salida se está descartando. Gracias a esta función logramos escribir un archivo dentro de la carpeta proc.

- task_struct: cada estructura de datos task_struct describe un proceso o tarea en el sistema. Para la creación de este módulo se utilizaron dos struct con task_struct una denominada task que por medio de un ciclo foreach, se logró obtener su id, nombre y estado, gracias a los atributos pid, comm, state respectivamente. Seguidamente al obtener estos datos damos inicio a otro ciclo foreach, aca utilziamos el segundo struct denominado task_child, el cual nos permitirá obtener cada uno de los procesos hijos del padre obtenido en el proceso anterior.

# INSTALACION DEL KERNEL (Proyecto)

- Descargar o clonar la información del repositorio: https://github.com/soylmml/SO1_Proyecto1
- Ya teniendo el repositorio en nuestro ordenador nos ubicamos en la carpeta memo_201314474_201314160, luego abrimos una terminal en esta carpeta, y procedemos a compilar los archivos de esta carpeta con el comando **make all.**
- Procedemos a instalar el modulo con la siguiente línea de código **sudo insmod memo_201314474_201314160.ko.**
- Procedemos a verificar el mensaje de inserción con el comando **dmesg.**
- Para verificar que nuestro modulo fue instalado correctamente procedemos a ubicarnos en la carpeta proc, dentro de la carpeta proc corremos el comando **cat memo_201314474_201314160,** y automáticamente podremos visualizar la información solicitada en la creación de este módulo.
- Procedemos a ubicarmos en la carpeta cpu_201314474_201314160, luego abrimos una terminal en esta carpeta, y procedemos a compilar los archivos de esta carpeta con el comando **make all.**
- Procedemos a instalar el modulo con la siguiente línea de código **sudo insmod cpu_201314474_201314160.ko.**
- Procedemos a verificar el mensaje de inserción con el comando **dmesg.**
- Para verificar que nuestro modulo fue instalado correctamente procedemos a ubicarnos en la carpeta proc, dentro de la carpeta proc corremos el comando **cat cpu_201314474_201314160,** y automáticamente podremos visualizar la información solicitada en la creación de este módulo.