Universidad De San Carlos De Guatemala
Facultad De Ingeniería
Escuela De Ciencias Y Sistemas
Sistemas Operativos 2 Sección A
Ing. Edgar René Ornelis Hoil
Tutor 1: Derek Esquivel Díaz
Tutor 2: Josue Rolando Gramajo Roldán
Segundo Semestre 2024

Practica 2

Multithreading

Objetivos

- Comprender como funcionan los hilos en Linux.
- Comprender los conceptos de concurrencia y paralelismo.
- Aplicar conceptos de sincronización de procesos.

Descripción

Usted ha sido seleccionado para formar parte del desarrollo de USAC Linux, esta será una distribución ligera de Linux desarrollada por estudiantes de ingeniería de la Universidad de San Carlos de Guatemala.

Uno de los directivos de la facultad solicitó al equipo de desarrollo que la distribución tuviera especial atención a la seguridad, por lo que solicito que esta incluyera un mecanismo de encriptación y desencriptación de archivos.

Los estudiantes implementarán llamadas al sistema que realicen cifrado y descifrado XOR de archivos de texto en paralelo.

Especificaciones

Se deberán de crear 2 llamadas al sistema; *my_encrypt* y *my_decrypt*

La llamada al sistema *my_encrypt* requerirá 4 parametros:

- ruta_archivo_entrada: Ruta del archivo a encriptar.
- ruta_archivo_salida: Ruta del archivo encriptado.
- número_de_hilos: Número de hilos con los que se encriptará el archivo.
- ruta_archivo_clave: Ruta del archivo que contiene la clave de encriptación.

El archivo de entrada se divide en n fragmentos (donde n es el número de hilos especificados). Cada fragmento se cifra de forma independiente mediante cifrado XOR con la clave proporcionada. Luego, los fragmentos cifrados se vuelven a escribir en el archivo de salida en el mismo orden.

La llamada al sistema *my_decrypt* requerirá 4 parametros:

- ruta_archivo_entrada: Ruta del archivo a desencriptar.
- ruta_archivo_salida: Ruta del archivo desencriptado.
- número_de_hilos: Número de hilos con los que se desencriptará el archivo.
- ruta_archivo_clave: Ruta del archivo que contiene la clave de encriptación.

El archivo cifrado se divide en los mismos fragmentos n que durante el cifrado. Cada fragmento se descifra igualmente de forma independiente mediante el descifrado XOR con la misma clave y número de hilos utilizados durante el cifrado. Luego, los fragmentos descifrados se vuelven a ensamblar y se escriben en el archivo de salida en el orden correcto.

Funcionamiento

División de fragmentos y asignación de subprocesos:

- El archivo de entrada se lee en la memoria y se divide en n fragmentos del mismo tamaño (o lo más iguales posible).
- Cada fragmento se asigna a un hilo independiente para su procesamiento.
- Por ejemplo, si el archivo tiene 10,000 bytes y utiliza 4 hilos, cada subproceso manejaría 2,500 bytes.
- Por otro lado, si el archivo no puede ser dividido en partes exactas el ultimo fragmento será más grande para compensar, por ejemplo, si el archivo tiene 5,000 bytes y se quieren utilizar 3 hilos, los fragmentos 1 y 2 tendrán 1666 bytes mientras que el 3 tendrá 1668 bytes.

Cifrado XOR paralelo:

- Cada hilo aplica cifrado XOR a su porción de datos asignada usando la clave proporcionada.
- Después del cifrado, los hilos se sincronizan para garantizar que todos los fragmentos se procesen antes de combinar los resultados.
- Luego, los fragmentos se escriben secuencialmente en el archivo de salida.

Descifrado XOR paralelo:

- El proceso de descifrado refleja el proceso de cifrado. El archivo se divide en la misma cantidad de fragmentos y se procesa mediante la misma cantidad de hilos.
- Cada hilo aplica el descifrado XOR a su fragmento utilizando la misma clave.
- Los fragmentos descifrados se combinan y se vuelven a escribir en el archivo de salida.
- Asegúrese de que la cantidad de hilos utilizados en la llamada al sistema my_decrypt coincida con el número utilizado durante my_encrypt. De lo contrario, el descifrado producirá resultados incorrectos.

Comprobación

Desarrollar dos aplicaciones de espacio de usuario con C (encrypt_app y decrypt_app) que analicen argumentos de línea de comandos, invoquen llamadas al sistema y manejen rutas y parámetros de archivos. Estas apps deberán de ser llamadas de la siguiente manera:

encrypt_app

-p <ruta_archivo_entrada>
-o <ruta_archivo_salida>
-j <número_de_procesos>
-k <ruta_archivo_clave>

decrypt_app

-p <ruta_archivo_entrada>
-o <ruta_archivo_salida>
-j <número_de_procesos>
-k <ruta_archivo_clave>

Todos los argumentos serán obligatorios para correr la aplicación, de faltar uno se deberá mostrar un error.

Las aplicaciones no podrán tener código extra, su única función será la de leer los argumentos del usuario y ejecutar la llamada al sistema creada anteriormente.

Documentación

Trabajar con el kernel es algo complejo y difícil de entender si no se tiene el conocimiento adecuado, es por eso por lo que se le solicitará que escriba un informe detallado que documente los pasos seguidos en toda la práctica, así como los problemas con los que se encontró (o que observo que son comunes) y las soluciones encontradas (al menos 3).

Es importante que para esta documentación proporcione fragmentos de código y explicaciones para las modificaciones al kernel y la implementación de las llamadas al sistema.

Observaciones

- La práctica se realizará de manera individual.
- El lenguaje de programación a utilizar será C.
- La documentación se realizara en Markdown **(no se aceptarán otros formatos).**
- Se deberá trabajar sobre lo realizado durante la practica 1, tanto para los archivos modificados en el kernel como para la documentación.
- Las llamadas al sistema deberán de ser desarrolladas en su propio archivo “syscalls_usac.h” y ser compiladas junto al kernel, no se aceptará que estén incluidas en otro archivo como sys.h.

Entregables

- Código fuente **únicamente** de los archivos del kernel modificados por el estudiante, respetando la estructura de los directorios.
    - Por ejemplo, si el estudiante modifico include/linux/sys.h, deberá crear las carpetas include y linux.
- Documentación.

Forma de entrega

- Esta práctica se trabajará en el repositorio de GitLab de la clase donde se trabajó la practica 1.
    - Si no fue agregado durante la practica 1 comunicarse con su auxiliar.
- La entrega se realizará por medio de UEDI en el apartado correspondiente, donde el estudiante subirá un archivo de texto con el link de su rama en el repositorio.
- Tener especial cuidado de no modificar el trabajo de otros compañeros. De detectarse que modifico o copio el trabajo de alguien más será penalizado.

La entrega se debe realizar antes de las 23:59 del 19 de septiembre del 2024.