SISTEMAS OPERATIVOS

PIPES AND FIFOS

DATOS DEL ALUMNO:

- **NOMBRE DE ALUMNO**: FRANCO REYES
- **CARRERA:** INGENIERÍA EN SISTEMAS DE INFORMACIÓN
- **NÚMERO DE LEGAJO:** 10911

Respaldo teórico para resolución de actividades Pipes y FIFOs

¿Qué es un FIFO?

Un FIFO (First-In, First-Out) es un tipo de archivo especial en Linux llamado named pipe. Permite la comunicación entre procesos, donde lo que un proceso escribe otro lo puede leer. Se crea con mkfifo o mediante la función mkfifo() en C.

Comunicación Bidireccional con Pipes

Un proceso padre envía datos a un hijo usando un pipe. El hijo los transforma (por ejemplo, a mayúsculas) y los devuelve al padre por otro pipe. Esto requiere 2 pipes y cerrar extremos no utilizados en cada proceso.

Persistencia del número de secuencia

Para que un servidor FIFO recuerde el último número asignado, se guarda en un archivo. Cuando el servidor inicia, lee ese valor del archivo (si existe) y continúa desde ahí.

Manejo de señales

Cuando un servidor FIFO recibe una señal como SIGINT, debería liberar recursos como eliminar su archivo FIFO antes de finalizar. Esto se hace con una función manejadora conectada con signal().

Reapertura del FIFO en vez de mantenerlo abierto

Si el servidor cierra el FIFO después de cada lectura y lo vuelve a abrir, puede bloquearse. Es mejor mantenerlo abierto con un segundo descriptor para evitar EOF.

Protección contra clientes maliciosos

Si un cliente no abre su FIFO para lectura, el servidor puede quedar bloqueado al escribir. Se debe usar O_NONBLOCK o un timeout para evitarlo.

I/O no bloqueante

Con O_NONBLOCK, una operación de open() o read()/write() no bloquea. Permite que un servidor sea más robusto ante errores o bloqueos inesperados.

DATOS DEL ALUMNO:

- **NOMBRE DE ALUMNO**: FRANCO REYES
- **CARRERA:** INGENIERÍA EN SISTEMAS DE INFORMACIÓN
- **NÚMERO DE LEGAJO:** 10911

Ejemplo en python

El servidor del archivo fifo_seqnum_server.py realiza una segunda apertura de escritura sobre su FIFO para evitar que se cierre (EOF) cuando no hay clientes conectados.

Para resolver esto debemos hacer lo siguiente:

Modificar el comportamiento del servidor para que no mantenga un descriptor de escritura abierto artificialmente.

FLUJO DEL SERVIDOR (HECHO EN PYTHON)

El nombre del archivo es fifo_seqnum_server

Crea el FIFO

- El servidor verifica si el archivo especial FIFO ya existe.
- Si no existe, lo crea con os.mkfifo(FIFO_PATH).
- Esto crea un canal especial en disco (por ejemplo, en /tmp/mi_fifo) que permite comunicación entre procesos.

Entra en un bucle infinito

- Usa while True: para quedarse funcionando de forma continua.
- Así, el servidor nunca se detiene y está listo para atender mensajes todo el tiempo.

Abre el FIFO para lectura

- Usa open(FIFO_PATH, 'r') para esperar que alguien lo abra en modo escritura.
- ⚠️ Este paso bloquea (se queda esperando) hasta que un cliente lo abra para escribir.

Lee los mensajes

- Una vez abierto, comienza a leer línea por línea con un for line in fifo: (o readline()).
- Cada línea es un mensaje enviado por el cliente.

DATOS DEL ALUMNO:

- **NOMBRE DE ALUMNO**: FRANCO REYES
- **CARRERA:** INGENIERÍA EN SISTEMAS DE INFORMACIÓN
- **NÚMERO DE LEGAJO:** 10911

- El servidor procesa o imprime esa línea.

Detecta EOF si el cliente se va

- Si el cliente termina o cierra su lado de escritura, el servidor recibe EOF (fin de archivo).
- Esto significa que no hay más datos para leer.
- El servidor entonces cierra el FIFO y lo vuelve a abrir en la próxima vuelta del bucle → vuelve a esperar un nuevo cliente.

Ctrl+C (KeyboardInterrupt) → Cierra y limpia

- Si el usuario presiona Ctrl+C, se lanza una excepción KeyboardInterrupt.
- El servidor entra al bloque except y:
  - Imprime un mensaje de salida.
- Elimina el FIFO del sistema de archivos (os.remove()), para dejar limpio el entorno.