SISTEMAS OPERATIVOS AVANZADOS (TEORÍA)[1]

VIERNES 19 NOVIEMBRE 2021

1. ¿Qué abstracciones emplea el OS para virtualizar la CPU y la Memoria? ¿Por qué requieren de algún soporte por parte del hardware para ser empleadas?

2. ¿Por qué el PCB de un proceso no incluye el `%eip`? ¿Cómo se restaura en el cambio de contexto el `%eip` del proceso entrante?

3. ¿Por qué la línea (en lenguaje de alto nivel) inmediatamente posterior a syscall `exec()`, en condiciones normales, nunca se ejecuta?

4. ¿Por qué las colas de más baja prioridad el MLFQ tienen un quantum de tiempo mayor?

5. ¿En que condiciones para ejecutar el código de una llamada al sistema no se requiere modificar el espacio de direcciones del proceso (i.e., modificar el registro del procesador que mantiene del puntero base de la tabla de paginas `%cr3`?

6. Suponiendo que tenemos un ISA con 52 bits de direccionamiento ¿Cuántos niveles en la tabla de páginas emplearías? ¿Cuál sería el tamaño de los marcos de página?

7. ¿A que algoritmo de planificación (de los vistos) se asemeja más al planificador en el *mainline* del kernel de Linux? ¿Por qué dicho planificador usa un red-black-tree?

8. Tras un fallo de página (asumiendo un subsecuente remplazo), ¿Qué operación hay que hacer en el TLB y por qué?

9. ¿Cuántos accesos a memoria implica ejecutar la instrucción `movl $0x0,%eax` en un procesador x86, suponiendo que el sistema operativo emplea paginación en dos niveles y todas las entradas del TLB son inválidas?

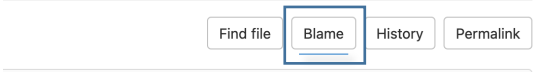
10. ¿Por qué la duración del quantum de planificación suele ser mayor que la periodicidad del timer?

[1] Todas las respuestas han de ser justificadas adecuadamente

SISTEMAS OPERATIVOS AVANZADOS (PRÁCTICAS)

VIERNES 19 NOVIEMBRE 2021

Para cada una de las prácticas hay que explicar que comprueba el test indicado y explicar de que modo lo resuelve **tú implementación** de la práctica (fichero/s y líneas de código). Indicar el *commit* que incluye el cambio (8 primeras cifras del hash). Ser recomienda resolver el examen empleado exclusivamente el interfaz de ***gitlab***. Cuando se explora el fichero de interés, y hacéis *click* en "*Blame*" en el menú superior



En la columna izquierda sale el *commit* de cada línea de código. Al hacer *click* os lleva al *commit* correspondiente al cambio.

PRACTICA 1

test/P1/ctest/proc3.c

PRÁCTICA 2

test/P1/ctest/setpri.c
test/P1/ctest/awake_high_priority.c

PRÁCTICA 3

tests/P3/ctests/shmem_access_full_address_space.c
tests/P3/ctests/bounds.c