PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

SISTEMAS OPERATIVOS

4ta práctica (tipo a)
(Primer semestre de 2023)

Horarios 0781, 0782

Duración: 1 h. 50 min.
Nota: No se puede usar ningún material de consulta.
La presentación, la ortografía y la gramática influirán en la calificación.
Puntaje total: 20 puntos

Pregunta 1 (3 puntos – 15 min.) Consider a file system with 2048 byte blocks and 32-bit disk and file block pointers. Each file has 12 direct pointers, a singly-indirect pointer, a doubly-indirect pointer, and a triply-indirect pointer.

a) (1 punto) How large of a disk can this file system support? Exprésalo en bytes con el prefijo correspondiente. La respuesta final no debe ser expresada en las potencias de 2.

b) (2 puntos) What is the maximum file size? Exprésalo como suma de valores con prefijos correspondientes. La respuesta final no debe ser expresada en las potencias de 2.

Pregunta 2 (3 puntos – 15 min.) List the set of disk blocks that must be read into memory in order to read the file /home/inf239/test.txt in its entirety from a UNIX BSD 4.2 file system (10 direct pointers, a singly-indirect pointer, a doubly-indirect pointer, and a triply-indirect pointer). Assume the file is 15,234 bytes long and that disk blocks are 1024 bytes long. Assume that the directories in question all fit into a single disk block each. Note that this is not always true in reality.

Pregunta 3 (4 puntos – 20 min.) Como se sabe, en los sistemas Unix todo son archivos, y cada archivo tiene su *inode* único. El valor de *inode* es parte de la estructura `stat` que se obtiene con la función `fstat` indicando el descriptor del archivo en interés. En el siguiente programa se imprimen algunos *inodes*. Su tarea es entender el código del programa, presentar el árbol de los procesos (1 punto), las comunicaciones entre los procesos (1 punto) y la salida que produce el programa con unos valores arbitrarios pero relacionados (1 punto). Si la salida se presenta en el orden correcto, usted obtiene 1 punto más.

```
$ cat pipes_inodes.c
#include <sys/stat.h>
#include <stdlib.h>
#include <stdio.h>

#define n 3

int
main(void)
{
  int fd[2], i=0, p;
  struct stat sb;

  pipe(fd); dup2(fd[0],0); dup2(fd[1],1); close(fd[0]); close(fd[1]);
  while (i < n) {
    pipe(fd);
    p=fork();
    p ? dup2(fd[1],1) : dup2(fd[0],0);
    close(fd[0]); close(fd[1]);
    if (p) break;
    i++;
  }
  while ( waitpid(-1, NULL, NULL) != -1 );
  fprintf(stderr, "Process %d. inodes: ", getpid());
  fstat(0, &sb);
  fprintf(stderr, "0(%ld), ", sb.st_ino);
  fstat(1, &sb);
  fprintf(stderr, "1(%ld)\n", sb.st_ino);
  exit(0);
}
$ gcc pipes_inodes.c -o pipes_inodes
$ ./pipes_inodes
...
```

Pregunta 4 (10 puntos – 50 min.) En MS Windows, al intentar eliminar un archivo abierto aparece la ventana del error "Archivo en uso" informando que "La acción no se puede completar porque la aplicación tiene abierto el archivo. Cierre el archivo e inténtelo de nuevo." En los sistemas de archivo con *i-nodes* el comportamiento en este caso es diferente:

```
$ date >foo
$ ls -l foo
-rw-r--r-- 1 vk vk 29 jun 10 00:44 foo
$ gedit foo &
[2] 13913
$
[2]-  Hecho                   gedit foo        # el editor tiene la ventana abierta
$ rm foo
$ ls -l foo
ls: no se puede acceder a foo: No existe el archivo o el directorio
# al cerrar la ventana, aparece el mensaje "¿Guardar los cambios del documento «foo» antes de cerrar?"
# si se guarda ...
$ ls -l foo
-rw-r--r-- 1 vk vk 29 jun 10 00:48 foo
```

(*By Michael Kerrisk*) "… In addition to maintaining a link count for each i-node, the kernel also counts open file descriptions for the file. If the last link to a file is removed and any processes hold open descriptors referring to the file, the file won't actually be deleted until all of the descriptors are closed. This is a useful feature, because it permits us to link a file without needing to worry about whether some other process has it open. (However, we can't reattach a name to an open file whose link count has fallen to 0). In addition, we can perform tricks such as creating an opening a temporary file, unlinking it immediately, and then continuing to use it within our program, relying on the fact that the file is destroyed only when we close the file descriptor – either explicitly, or implicitly when the program exits."

a) (3 puntos – 15 min.) Considerando la propiedad descrita, explique qué sucedería con los componentes del sistema de archivo (la entrada del directorio, el *inode*, el bloque del archivo `foo`) durante la ejecuciones de todas las órdenes presentadas anteriormente (menos `ls`). Indique como se organiza la relación entre ellos. También indique qué llamadas al sistema se usan en cada orden.

Ahora considere el siguiente programa:

```
$ cat t_unlink
#include <sys/stat.h>
#include <fcntl.h>
#include "tlpi_hdr.h"

#define CMD_SIZE 200
#define BUF_SIZE 1024

int
main(int argc, char *argv[])
{
    int fd, j, numBlocks;
    char shellCmd[CMD_SIZE];
    char buf[BUF_SIZE];

    if (argc < 2 || strcmp(argv[1], "--help") == 0)
        usageErr("%s temp-file [num-1kB-blocks] \n", argv[0]);

    numBlocks = (argc > 2) ? getInt(argv[2], GN_GT_0, "num-1kB-blocks")
                           : 100000;

    fd = open(argv[1], O_WRONLY | O_CREAT | O_EXCL, S_IRUSR | S_IWUSR);
    if (fd == -1) errExit("open");

    if (unlink(argv[1]) == -1) errExit("unlink");

    for (j = 0; j < numBlocks; j++)
        if (write(fd, buf, BUF_SIZE) != BUF_SIZE) fatal("partial/failed write");

    snprintf(shellCmd, CMD_SIZE, "df -k `dirname %s`", argv[1]);
    system(shellCmd);
    if (close(fd) == -1) errExit("close");
    printf("********** Closed file descriptor\n");
    system(shellCmd);
    exit(EXIT_SUCCESS);
}
```

```
$ ./t_unlink /tmp/tfile 1000000
Filesystem           1K-blocks       Used Available Use% Mounted on
/dev/sda10             5245020    3204044   2040976  62% /
********** Closed file descriptor
Filesystem           1K-blocks       Used Available Use% Mounted on
/dev/sda10             5245020    2201128   3043892  42% /
```

b) (4 puntos – 20 min.) Explique cuál es el objetivo del programa y comente las operaciones para lograr este objetivo durante su ejecución.

c) (3 puntos – 15 min.) Supongamos que el archivo `tfile` se crea en el sistema de archivo ext2 (un número de bloque en el *inode* ocupa 4 bytes) con los bloques de 4KiB, ¿se usaría el bloque indirecto doble para este archivo? ¿El indirecto triple? ¿Aproximadamente, cuántos bloques administrativos se usarían para los bloques de datos del archivo `tfile`?

La práctica ha sido preparada por VK
en Linux Mint 21.1 "Vera" con LibreOffice Writer.

Profesor del curso: V. Khlebnikov

Pando, 22 de junio de 2023