Ejercicio de Ficheros Junio 2012

Considere un sistema de ficheros UNIX estándar con las siguientes características: tamaño de bloque: 4KB, tamaño del i-nodo (sólo información, sin punteros): 128 bytes, tamaño del puntero (a i-nodo o a bloque): 8 bytes, punteros dentro del i-nodo: directos 13, indirectos 3 (simple, doble y triple, respectivamente) y entradas de directorio estáticas: 120 caracteres y 1 puntero al i-nodo.

a) ¿Cuál es el tamaño máximo del fichero direccionable por este formato de i-nodo?
b) Si se desean almacenar en un mismo directorio 100 ficheros de 4MB:
   - ¿Cuántos i-nodos se consumen? Tanto por el directorio como por los ficheros.
   - ¿Cuántos bloques de datos se consumen? Por el directorio y por los ficheros.

El concepto de sistemas de ficheros con versiones (*versioning file system*) incluye la posibilidad de mantener un acceso a copias antiguas de un mismo fichero de una forma transparente. Cada una de estas copias está asignada a un instante de tiempo (*snapshot*) determinado. Un ejemplo de uso sería:

```
[user@machine] echo "Versión original del fichero foo.txt" > foo.txt
[user@machine] snapshot
Snapshot on . 1893477
[user@machine] echo "Versión actualizada del mismo fichero foo.txt" > foo.txt
[user@machine] ls
.    ..    foo.txt
[user@machine] cat foo.txt@1893477
Versión original del fichero foo.txt
[user@machine] cat foo.txt
Versión actualizada del mismo fichero foo.txt
```

c) [2½ puntos] Existen varias posibles alternativas para que un mismo nombre de fichero (dependiendo del snapshot) haga referencia a uno u otro contenido. Si se quisiese mantener la misma estructura de i-nodo y sólo modificar el formato de la entrada de directorio. ¿Qué modificaciones de diseño habría que hacer? ¿Se tendría que modificar la implementación de alguno de los siguientes servicios del sistema operativo: *readdir*, *unlink* (borrado de fichero) y *creat*?

   Considere que el contador de *snapshot* es un entero de 8 bytes.

d) [1 punto] Con esa modificación de diseño ¿Sería posible hacer un enlace simbólico de un fichero de un *snapshot* anterior?

```
[user@machine] ln -s foo.txt@1893477 old-foo.txt
```

   ¿Y un enlace físico?

Si en el caso anterior de los 100 ficheros de 4MB, entre un *snapshot* y otro, se modifican los últimos 20 bytes de cada fichero.

e) Para mantener las dos versiones, ¿cuántos i-nodos se consumen? Tanto por el directorio como por los ficheros. ¿Cuántos bloques de datos se consumen? Por el directorio y por los ficheros.

Algunas implementaciones de sistemas de ficheros con versiones implementan la funcionalidad *copy-on-write*, que permite que los bloques de un fichero que no se hayan cambiado entre una versión y la siguiente se compartan entre ambos ficheros.

f) ¿Incluir esta modificación alteraría la estructura del i-nodo o de los bloques indirectos? ¿De qué manera? Diseñe esa nueva estructura.
g) ¿Cuánto ocuparían ahora, en bloques de datos, uno de los ficheros de 4MB? ¿Y cuánto espacio más requerirá la versión en la que hay modificados los últimos 20 bytes?

Solución

a) [1 punto] ¿Cuál es el tamaño máximo del fichero direccionable por este formato de i-nodo?

El i-nodo tiene hasta un puntero de indirección triple, para conocer el número de bloques direccionables es necesario calcular cuántos punteros a bloque caben en cada uno de los bloques de indirección:

Tam_Bloque / Tam_puntero = 4KB / 8 bytes = 2^{12} bytes / 2^3 bytes = 2^9 punteros = 512 punteros

Entonces los bloques direccionables son:

- Directos: 13 bloques
- Indirecto simple: 2^9 bloques
- Indirecto doble: $2^9 x 2^9 = 2^{18}$ bloques
- Indirecto triple: $2^9 x 2^9 x 2^9 = 2^{27}$ bloques

El total en KBs (a 4KB por bloque): $(13+2^9+2^{18}+2^{27}) \times 4KB \approx 2^{27} x 2^{12}$ bytes = 2^{39} bytes =0.5 TB (en el orden de medio terabyte).

b) [1½ puntos] Si se desean almacenar en un mismo directorio 100 ficheros de 4MB:

- **[½ puntos] ¿Cuántos i-nodos se consumen? Tanto por el directorio como por los ficheros:**
  Se consume 1 i-nodo por fichero más 1 i-nodo por directorio: 101 i-nodos.

- **[1 punto] ¿Cuántos bloques de datos se consumen? Por el directorio y por los ficheros:**
  Un fichero de 4MB ocupa 4MB / 4KB/bloque= 1024 bloques.
  Para indicar esos 1024 bloques, hacen falta:
  - Los 13 punteros directos.
  - 1 puntero indirecto simple que a su vez direcciona 2^9 bloques=512 bloques
  - 1 puntero indirecto doble, que apunte a un segundo i-nodo de indirección y de ahí a los bloques restantes: 1024-13-512=499 bloques

  En total (por fichero): 1024 bloques de datos + 1 indirecto simple + 1 indirecto doble + 1 i-nodo del segundo nivel de indirección = 1027 bloques.
  En ficheros ocupamos 100 ficheros x 1027 bloques/fichero = 102700 ficheros.

  Ahora, el directorio: Éste tiene 100 entradas de ficheros + 2 entradas (correspondientes al "." y ".."), en total 102 entradas de directorio.

  Cada entrada ocupa 120 carcateres (bytes) + 8 bytes de dirección de i-nodo=128 bytes.

  En un bloque de disco entran 4KB/bloque / 128 bytes/entrada = 2^{12} / 2^7 entradas/bloque= 2^5 entradas/bloque = 32 entradas/bloque

  Para las 102 entradas se necesitan 102 entradas / 32 entradas/bloque < 4 bloques (todos ellos direccionables por los punteros directos).

  En total 102700 bloques en ficheros + 4 bloques en contenidos de directorios=102704 bloques.

c) [2½ puntos] Existen varias posibles alternativas para que un mismo nombre de fichero (dependiendo del *snapshot*) haga referencia a uno u otro contenido. Si se quisiese mantener la misma estructura de i-nodo y sólo modificar el formato de la entrada de directorio.
[1 punto] ¿Qué modificaciones de diseño habría que hacer?

La modificación de diseño (sin alterar la estructura del i-nodo) debe afectar únicamente a la estructura del contenido del directorio, que actualmente es:

120 bytes	8 bytes
Nombre de la entrada	Nro. i-nodo

La nueva estructura tiene que contener entre que dos snapshots es válida una entrada de directorio,

120 bytes	8 bytes	8 bytes	8 bytes
Nombre de la entrada	Snapshot ini	Snapshot fin	Nro. i-nodo

Con esta estructura, el contenido del directorio para el ejemplo anterior sería:

`.`	0	-1	412
`..`	0	-1	520
`foo.txt`	0	1893477	812
`foo.txt`	1893478	-1	825

El valor "-1" en los snapshot quiere decir "hasta el instante actual", los números de i-nodo pueden ser cualquiera.

[1½ puntos] ¿Se tendría que modificar la implementación de alguno de los siguientes servicios del sistema operativo: *readdir*, *unlink* (borrado de fichero) y *creat*?

- *readdir*: Esta llamada recorre las entradas de directorio, por lo tanto, lo que tendrá que hacer es saltarse aquellas que se correspondan con el estado del mismo para el snapshot soliticado (que puede ser uno en concreto) o lo más habitual que sea para el instante actual (caso en el cual se tendrían que mostrar aquellas que tienen como "snapshot fin" un "-1".
- *unlink*: El borrado de un fichero, debe considerar dos casos: (i) que el fichero haya sido creado después del último snapshot, entonces se borra y nada más (funcionamiento estándar), y (ii) si el fichero existía en snapshots anteriores (es decir su "snapshot ini" es menor que el último snapshot) entonces se cambia el "snapshot fin" al valor del último snapshot realizado (indicando que dicho fichero existía entre esos dos snapshots).
- *creat*: La creación de un fichero implica el borrado de un fichero anterior que tuviese el mismo nombre, por lo tanto esa parte de borrado (si existe un fichero con ese nombre) sería isual que el caso de *unlink*. Para lo que es creación del fichero nuevo (exista o no uno anterior) se crearía una nueva entrada con el nombre correspondiente, se pondría en "snapshot ini" el valor del último *snapshot* realizado +1 y como "snapshot fin" el valor "-1".

d) [1 punto] Con esa modificación de diseño

[½ puntos] ¿Sería posible hacer un enlace simbólico de un fichero de un snapshot anterior?

Un enlace simbólico consiste en un fichero cuyo contenido es la referencia (mediante una ruta relativa o absoluta) de otro fichero dentro del sistema de ficheros. Si aceptamos la nomenclatura *fichero@snapshot* que hemos usado antes, no habría problema para que dicha referencia fuese la indicada dentro del enlace simbólico.

[½ puntos] ¿Y un enlace físico?

Tampoco, ya que sería una entrada del directorio que apuntaría al mismo número de i-nodo.

En único problema que se podría dar en ambos casos es qué pasaría con las escritoras, ya que se estaría cambiando el contenido de una copia versionada de un *snapshot* anterior. La solución de compromiso más razonable es hacer que tanto enlaces simbólicos como físicos a ficheros antiguos no tengan permisos de escritura.

e) [1½ puntos] Para mantener las dos versiones:

Esta opción de diseño tratada aquí, implica que un fichero modificado después de hacer un snapshot, implica la creación de un nuevo fichero con los contenidos actualizados, por lo tanto:

- **[½ puntos] ¿cuántos i-nodos se consumen? Tanto por el directorio como por los ficheros.**
  I-nodos consumidos: Para los ficheros, el doble 100 i-nodos x 2 = 200 i-nodos + i-nodo del directorio.
- **[1 punto] ¿Cuántos bloques de datos se consumen? Por el directorio y por los ficheros.**
  Bloques de datos: Para empezar, todos los bloques de datos de los ficheros se duplican 102700 bloques x 2 = 205400 bloques.

Los bloques de para el contenido de directorio se ven afectados de doble manera:

- Ahora el tamaño de las entradas es de 120 bytes (nombre) + 8 bytes (snapshot ini) + 8 bytes (snapshot fin)+ 8 bytes (i-nodo) = 144 bytes. Por lo tanto en un bloque de datos entran sólo 4KB / 144 bytes = 4096/144 entradas ≈ 28 entradas.
- El número de entradas a incluir ahora es mayor, ya que de cada fichero tendremos la entrada asociada a la versión antigua y la de la versión nueva: 2 entradas ("." y "..") + 2x 100 entradas de ficheros = 202 entradas.

En total se necesitarían 202 entradas / 28 entradas/bloque ≈ 8 bloques para contenidos de directorios.
En total 205400 bloques de ficheros + 8 bloques de directorios = 205408 bloques.

f) [1½ puntos] ¿Incluir esta modificación alteraría la estructura del i-nodo o de los bloques indirectos? ¿De qué manera? Diseñe esa nueva estructura.

El concepto de COW (*copy-on-write*) en un sistema de ficheros es el mismo que en el caso de las páginas de memoria (en un esquema con paginación) tras una llamada *fork*. En nuestro caso el momento en el que se crea un duplicado es cuando se modifica un fichero entre *snapshots*. En este caso, a diferencia de lo contado en el apartado anterior, cuando se crea el nuevo fichero, se crea una nueva entrada en el directorio y un nuevo i-nodo, pero no se duplican todos los bloques de datos, en lugar de ello se etiquetan como COW, de forma que sólo cuando se vaya a hacer una operación de escritura sobre ese bloque se duplicaría.

Esto implica que resultará necesario disponer de un campo más por puntero de bloque, ahí hay varias opciones: (i) ya que 8 bytes (64 bits) de tamaño de direcciones de bloque es muy grande (permite direccionar 2^{64} bloques, es decir hasta $2^{64}x2^{12}=2^{76}$ bytes = 64 ZB [Zettabytes]) debido a lo cual podríamos prescindir alguno de los bits de direccionamiento para indicar si el bloque es COW o no; (ii) la otra opción sería usar un campo (que por cuestiones de alineamiento debería de ser de mínimo un byte) para denotar la propiedad COW del bloque.

La primera de las alternativas no incrementa el tamaño necesario para mantener las estructuras de direccionamiento, pero sí que es una opción de bajo nivel poco elegante y que complica en una pequeña medida la gestión. La segunda alternativa, por contra, es más elegante y limpia pero repercute en aprovechamiento de las estructuras, por ejemplo, en un bloque de indirección en lugar de 2^{12} bytes / 2^3 bytes = 2^9 punteros = 512 punteros, ahora los punteros ocupan un byte más y tendríamos 2^{12} bytes / $(1+2^3)$ bytes = 455 punteros; también afectaría al i-nodo para el cual todos los punteros directos e indirectos ocuparían 1 byte más (haciendo crecer el tamaño de los i-nodos o reducir el número de punteros directos, por ejemplo).

g) [1 punto] ¿Cuánto ocuparían ahora, en bloques de datos, uno de los ficheros de 4MB? ¿Y cuánto espacio más requerirá la versión en la que hay modificados los últimos 20 bytes?

Si tomamos la opción (i) el espacio ocupado por los ficheros originalmente sería el mismo, ver apartado a), 1027 bloques.

Para la opción (ii), un fichero de 4MB ocupa 4MB / 4KB/bloque= 1024 bloques.
Para indicar esos 1024 bloques, hacen falta:

- Los 13 punteros directos.
- 1 puntero indirecto simple que a su vez direcciona 455 bloques
- 1 puntero indirecto doble, que apunte a un segundo i-nodo de indirección y de ahí a los bloques restantes: 1024-13-455=556 bloques necesitando un bloque más en el segundo nivel de indirección.

En total (por fichero): 1024 bloques de datos + 1 indirecto simple + 1 indirecto doble + 2 i-nodos del segundo nivel de indirección = 1028 bloques.

La principal ventaja de COW se ve cuando se tienen que almacenar el fichero modificado. La modificación sólo afecta al último bloque, por lo tanto sólo este bloque de datos se ocuparía como extra. Como el bloque modificado (que ya no sería COW) es uno referenciado por una indirección doble,

los dos bloques de indirección anteriores también se verían alterados. Por lo tanto, se ocuparían sólo 3 bloques adicionales.