LFS (Log File System)

Las máquinas y los discos son cada vez mas rápidas.
Las cantidades de memoria disponibles son cada vez mayores, lo cual permite cachés de gran tamaño, aliviando las lecturas, perod de todas formas, por cuestiones de seguridad, hay que ecribir los cachés en disco.

En definitiva, el factor limitante de velocidad del software es la gran cantidad de accesos random a disco.
La idea de LFS es desvincular a las aplicaciones de la escritura a disco y reemplazar las pequeñas escrituras de tipo random por escrituras masivas secuenciales, pero manteniendo la posibilidad de recuperar los datos del disco de forma eficiente.

LFS divide el disco estáticamente en **segmentos** de tamaño fijo (normalmente ½ Mb).
El ordenamiento lógico de estos segmentos crea un único log continuo.

LFS mantiene las estructuras estructuras de Ext2. Tiene un superblock, los inodes tienen la misma información (incluyendo punteros indirectos, doble indirectos y triple indirectos).

LFS reúne varias páginas que necesiten ser escritas y se prepara para escribirlas en el disco en el próximo segmento libre. En este punto, ordena los bloques por número de bloque, les asigna direcciones de disco y actualiza los metadatos necesarios. Los bloques de metadatos modificados se toman junto con los bloques de datos y se escriben todos juntos en un segmento del disco.

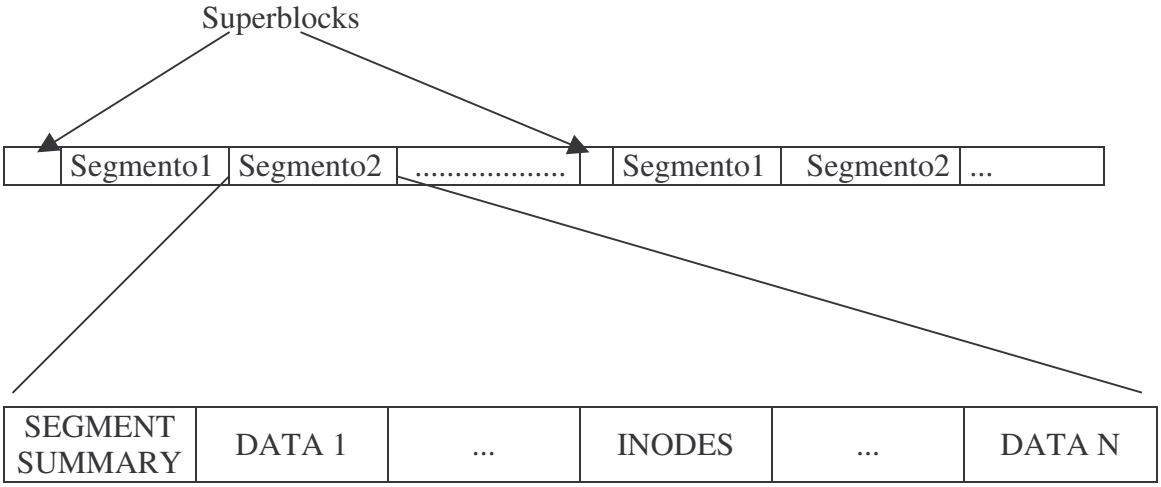
Checkpoint

Es una operación que provee el file system durante la cual, en un punto fijo en el disco se escriben todas las estructuras de datos del file system residentes en memoria durante una operación de Checkpoint. Un checkpoint marca un estado consistente del file system que puede ser usado para la recuperación después de un problema.
Tambien se escriben a disco todos los bloques de datos y metadatos modificados.
La política de recuperación consiste en recuperar el estado del file system a partir del último checkpoint. Todo lo hecho en el medio se pierde. Es posible regular el riesgo que esto implica aumentando o disminuyendo el intervalo de tiempo con el que se efectúan los checkpoints.
Aunque no esté implementado, LFS permite una mejor recuperación por medio del análisis del log (al igual que la mayoría de los motores de bases de datos). Una ventaja adicional es que le alcanza con revisar la cola del log desde el último checkpoint mientras que en los otros file systems es necesario recorrer todo el disco en busca de errores para permitir la recuperación.

Mapa de inodes
Como resultado de esta técnica, los inodes ya no están siempre en el mismo lugar del disco sino que su ubicación va cambiando conforme son actualizados.
Debido a esto, se necesita un ***Mapa de inodes*** que mapea un número de inode a su dirección en el disco.

Segmentos Parciales
Si los datos a escribir no alcanzan para completar un segmento pueden escribirse segmentos parciales

Estructura en Disco



Summary Checksum	
Data Checksum	
Puntero al próximo segmento	
Fecha y Hora de creación	
Numero de FInfos	Numero de inodes
FINFO 1	
. . .	
FINFO N	
. . .	
Inode disk address N	
. . .	
Inode Disk Address 1	

Numero de Bloques
Numero de Versión
Numero de Inode
Bloque lógico 1
. . .
Bloque lógico N

Cleaner
Los bloques modificados se escriben a disco en una posición diferente de la anterior. A esta técnica de reubicación se la llama política de no sobreescribir y necesita un mecanismo para recuperar el espacio resultante de bloques borrados o reescritos. Cleaner recorre los segmentos descartando aquellos bloques que ya no son válidos y agrega los nuevos. Para hacer esto necesita poder identificar los bloques dentro del segmento. esta información la obtiene del **Segment Summary.**