4.1. Interfaz de los sitemas de archivos

Archivos

Concepto de Archivo

Un **archivo** es la unidad de almacenamiento más pequeña, es una colección de información relacionada y almacenada en un dispositivo de memoria secundaria, en un espacio de direcciones lógicas contiguas.

Un archivo es una secuencia de bit, bytes, líneas o registros.

Los estructura del archivo depende del tipo:

- **Archivo de texto**: Caracteres organizada en líneas o páginas
- **Archivo fuente**: Subrutinas y funciones con instrucciones ejecutables
- **Archivo objeto**: Bytes organizados en bloques de forma que el programa pueda comprender
- **Archivo ejecutable**: Código que se puede cargar en memoria y ejecutar

Atributos de archivo (Metadatos)

- **Nombre**: única información en formato legible. Está compuesto por el nombre del archivo y la extensión que indica el tipo de archivo
- **Tipo**: Necesaria si el sistema soporta diferentes tipos de archivos
- **Identificador**: Normalmente un número que identifica al archivo dentro del sistema de archivos

- **Localización**: Puntero a un dispositivo y ubicación dentro del mismo
- **Protección**: Quién puede leer, escribir y ejecutar
- **Tamaño**: Tamaño actual y tamaño máximo
- **Fecha, hora e indentificación del usuario**: Necesarios para la protección, seguridad y monitorización del uso del archivo

Operaciones con archivos

Gestión:

- Crear
- Renombrar
- Copiar
- Borrar
- Establecer y obtener atributos

Procesamientos:

- Abrir y cerrar
- Leer
- Escribir (modificar, insertar, borrar información)
- Truncar (reinicializar la longitud a 0, sin modificar los atributos)

El sistema operativo mantiene la tabla de archivos abiertos, cuando realizamos una operación con un archivo se especifica mediante el índice de la tabla (descriptor de archivo) evitando recorrer así todo el directorio.

El *contador de aperturas* controla el número de aperturas y cierres del archivo. Así, si es cero la tabla de archivos abiertos se elimina.

El *puntero del archivo* es un puntero a la posición actual dentro del archivo, es distinto para cada proceso que opere dentro de un mismo archivo.

Tipos de archivos

- Regulares
- Directorios
- De dispositivos

Modo de acceso

- **Acceso Secuencial**: Método más simple. Se procesa en orden un registro tras otro. Escribe al final del archivo, dependiendo del sistema puede saltar *n* registros hacia delante o hacia atrás, generalmente se hace de uno en uno.
- **Acceso Directo**: un archivo compuesto de resgitros lógicos de longitud fija permiten acceder a dichos bloques sin ningún tipo de orden. Es muy útil para grandes cantidades de información.
- **Otros métodos de acceso**: Con índices que contienen los punteros a los distintos bloques. Si el archivo es grande solo se almacenan los índices en memoria principal.

Directorios

Estructura de almacenamiento

Una tabla contenidos de volumen o **directorio** de dispositivos tiene la información acerca de los archivos almacenados en el sistema, como, por ejemplo, el nombre, la ubicación, el tamaño y el tipo.

Tanto los directorios como los archivos residen en almacenamiento secundario.

Operaciones con Directorios

- Buscar un archivo
- Crear un archivo
- Borrar archivos
- Listar un directorio
- Renombrar archivos
- Recorrer el sistema de archivos

Organización de los directorios

La organización de los directorios debe proporcionar:

- **Eficiencia**: Localización rápida del archivo
- **Denominación**: Un mismo archivo puede tener diferentes nombres o varios usuarios pueden darle el mismo nombre a diferentes archivos

- **Agrupación**: Agrupar los archivos según sus propiedades

    - **Árbol**: Este tipo de estructura permite crear subdirectorios y organizar los archivos. Tiene un directorio raíz y todos los archivos tienen un nombre de ruta distintivo (absoluto o relativo).

      Búsquedas más eficientes.

      Cada proceso tiene un directorio actual debe de contener los archivos que actualmente interesen al proceso. Si estos archivos no se encuentran en este directorio habrá que indicarlo.

    - **Grafo**: Compartición de subdirectorios y de archivos. Esto hace que la estructura sea más flexible y compleja.

Compartición de archivos

- **Múltiples usuarios**: El propietario del archivo es el que puede cambiar los atributos del mismo para dar permiso de acceso a los demás usuarios o grupos.
- **Sistemas de archivos remotos**: Sistema de archivos distribuido (directorios remotos visibles desde una máquina local), montar más de un sistema de archivos.
    - **Cliente-Servidor**: El servidor especifica los archivos disponibles en el directorio. El cliente puede ser identificado por el nombre de una red o una dirección IP, pero esto puede ser suplantado haciendo que clientes no autorizados accedan a datos de otro servidor.
    - **Sistemas de información distribuidos**: Más fácil de gestionar que los Cliente-Servidor puesto que proporcionan un acceso unificado a la información.
    - **Fallo**: Los sistemas de archivos locales incluyen fallos de disco. Los errores de los administradores del sistema pueden hacer que se pierdan archivos. Podrán recuperarse los fallos si mantenemos información del estado del cliente y del servidor.

Semánticas de consistencia

Criterio para evaluar todo Sistema de Archivos que soporte la compartición de archivos. Especifica cómo se pueden acceder simultáneamente a un archivo y cuándo las modificaciones de un usuario se observan por los demás.

- **Semántica de UNIX**

    - La escritura en un archivo es visible a todos los usuarios.
    - Los usuarios pueden compartir el puntero de la ubicación actual dentro del archivo. Así, si se modifica ese puntero afectará a todos.
- **Semántica de sesión (Sistema de archivos Andrew)**
    - La escritura en un archivo no es visible a todos los usuarios.
    - Una vez cerrado el archivo, solo tendrán los cambios realizados los usuarios que tuviesen la sesión iniciada.
    - No hay restricción a la planificación de los accesos.
- **Semántica de archivos inmutables**
    - El nombre y el contenido del archivo no podrán modificarse una vez compartidos. Sólo lectura.

Protección

Fiabilidad: se proporcionan copias duplicadas de los archivos, por si resultase ser destruido accidentalmente.

Los mecanismos de protección proporcionan un acceso controlado, esto es, limitación de los accesos a los distintos archivos.

Tipos de acceso

- Lectura.
- Escritura
- Ejecución
- Adición

- Borrado
- Listado

Lista de control de acceso: se asocia a cada archivo o directorio, en ella se especifica el nombre de usuario y el tipo de acceso.

Desventajas: problemas de espacio en usuarios individuales, la construcción de la lista es tediosa y la entrada del directorio es de tamaño variable lo que complica la gestión de espacio.

Para solucionar estos problemas los usuarios se clasifican en propietario, grupo y público (otros).

Otra técnica de protección consiste en asociar una contraseña a cada archivo. La desesventaja es que el usuario tendría que recodar muchas contraseñas, por tanto, este método resultaría poco práctico. Una vez descubierta la contraseña se tendrá acceso total a todos los archivos ("todo o nada").

Funciones básicas del Sistema de Archivos

- Tener conocimiento de todos los archivos del sistema
- Controlar la compartición y forzar la protección de archivos
- Gestionar el espacio del sistema de archivos: Asignación y liberación del espacio en disco
- Traducir las direcciones lógicas del archivo en direcciones físicas del disco: Los usuarios especifican las partes que quieren leer o escribir en términos de direcciones lógicas relativas al archivo

4.2. Diseño Software del Sistema de Archivos

Problemas a la hora de diseñar un sistema de archivos:

- Definir el aspecto del SA para el usuario, esto implica definir un archivo y sus atributos, operaciones permitidas sobre el archivo y la estructura de directorios.
- Definir algoritmos y estructuras de datos que permitan mapear el sistema lógico sobre los dispositivos físicos donde se almacenan.

Organización por niveles

- **Controladores de dispositivos**: Entrada comandos de alto nivel y salida instrucciones a bajo nivel
- **Control de E/S** (rutinas de tratamiento de interrupciones): Transferir información entre disco y memoria principal
- **Sistema básico de archivos**: operaciones de lectura y escritura en bloques físicos del disco
- **Módulo de organización de archivos**: Conoce los archivos, bloques lógicos y físicos, así que puede traducir las direciones lógicas y en físicas
- **Sistema lógico de archivos**: Gestiona información de la estructura de directorios y protección
- **Programas de aplicación**:

  Un bloque de control de archivos: Contiene información acerca del archivo; como el propietario, permisos y ubicación

Métodos de Asignación

Asignación Contigua

Cada archivo ocupa un conjunto de bloques contiguos en disco.

- **Ventajas**:

  Sencillo: solo está definida por la dirección de comienzo del primer bloque y la longitud del archivo.

  Acceso tanto *secuencial* (se lee el siguiente bloque al último que se haya hecho referencia) como *directo*.

- **Desventajas**:

  No se conoce inicialmente el tamaño y además el archivo *no puede crecer* si se encuentra entre dos archivos. Para solucionarlo tenemos dos posibilidades: enviar un mensaje de error al usuario o localizar un hueco de mayor tamaño, copiar el archivo en la nueva posición y liberar el espacio anterior, pero esto sule consumir bastante tiempo.

  Problema de asignación dinámica del espacio, satisfacer la solicitud de aumento del tamaño a partir de huecos libres. Esto produce *fragmentación externa*, a medida que se van eliminando archivos de memoria el espacio libre de memoria queda fragmentado, derrochando así la memoria ya que en ningún fragmento es lo suficientemente grande como para almacenar un archivo.

  Una solución al problema de la fragmentación externa es la *compactación*, consiste en copiar los archivos en un disco, así, el disco original se liberaría por completo dejando todo el espacio libre contiguo y, a continuación, volver a introducir los archivos. Este

método es muy ineficiente debido al tiempo que se necesita para realizar el procedimiento.

Para minimizar los problemas se usan *extensiones*, los bloques se registran mediante una dirección, un número de bloque y la dirección al primer bloque de la extensión (otro área de espacio contiguo). La fragmentación externa puede continuar.

Asignación lógica a física

$$Dirección\ lógica\ /\ Tamaño\ bloque\ de\ disco\ \rightarrow\ C\ (cociente),\ R\ (resto)$$ $$Número\ de\ bloque\ =\ C\ +\ Dirección\ de\ Comienzo$$ $$Desplazamiento\ =\ R$$

Asignación No Contigua - Enlazada

Cada archivo tiene un lista enlazada de bloques de disco, estos bloques pueden estar dispersos por el disco.

- **Ventajas**:

  El directorio tiene un puntero al primer y el último bloque del archivo, además, cada bloque tiene un puntero al siguiente bloque. Por tanto, solo *basta con almacenar el puntero al primer bloque*.

  No hay fragmentación externa: se puede utilizar cualquier bloque de la lista de bloques libres.

  No necesitamos saber el tamaño del archivo cuando lo creamos, el *archivo puede crecer* mientras haya bloques libres.

- **Desventajas**:

  Solo *acceso secuencial*. El acceso directo es muy poco efectivo

debido a que el acceso a un puntero requiere una lectura en disco.

Si el *tamaño de los punteros* es grande estamos desperdiciando memoria en lugar de almacenar infomación.

La solución a este problema es agrupar bloques (clusters), así, asignaremos clusters en lugar de bloques disminuyendo el porcentaje en memoria usado por los punteros.

Pérdida de fiabilidad: los archivos están enlazados por punteros y alguno de estos puede ser dañado haciendo que apunte a bloques libre o a otros archivos. Una solución a este problema es una lista doblemente enlazada (overhead), otra solución consiste en almacenar el nombre del archivo y el número de bloque relativo en cada bloque.

Asignación lógica a física:

$$Dirección\ lógica\ /\ (Tamaño\ bloque\ de\ disco\ -\ Tamaño\ dirección)\ \rightarrow\ C\ (cociente),\ R\ (resto)$$ $$Número\ de\ bloque\ =\ C$$ $$Desplazamiento\ =\ R\ +\ 1$$

Tabla de Asignación de Archivos (FAT)

Es una variación del método enlazado que almacena en una tabla cada bloque y el bloque siguiente.

La asignación de la tabla puede provocar un número significativo de accesos a disco, salvo que la tabla se encuentre en caché (más simple y eficiente).

Reserva una sección de disco al comienzo de la partición para la FAT.

Tiene una entrada por cada bloque del disco y está indexada por número de bloque de disco.

Para localizar el bloque solo se necesita leer en la FAT, optimizando así el acceso directo.

El problema que presenta es la pérdida de punteros, que se soluciona mediante una doble copia de la FAT.

Asignación No Contigua-Indexada

Todos los punteros a los bloquesestán agrupados en una dirección cncreta, el **bloque de índice**.

Cada directorio tiene la localización a este bloque índice y cada archivo tiene asociado su propio bloque índice. La entrada i-ésima del bloque índice apunta al bloque i-ésimo del archivo.

- **Ventajas**:

  Buen acceso directo.

  No hay fragmentación externa, puede utilizarse cualquier bloque de disco para aumentar el espacio del archivo.

- **Desventajas**:

  Desperdicio de espacio: se requiere espacio adicional para almacenar los punteros del bloque índice, generalmente mayor que el que se requiere para la asignación enlazada.

  Soluciones:

- Esquema enlazado: Cada bloque índice ocupa un bloque en disco, podemos leer y escribir de disco directamente. Para archivos más grandes podemos enlazar varios bloques índices.

- Índices multinivel: El bloque índice apunta al primer nivel, estos

apuntarán a los del segundo nivel y a su vez a los bloque del archivo. El problema es que hay que acceder a disco para recuperar la dirección del bloque para cada nivel de indexación. La solución es mantener algunos bloques en memoria principal.

- Esquema combinado (Unix): Los primeros punteros del bloque índice en el nodo del archivo. De estos, unos hacen referencia a bloques directos, es decir, contienen las direcciones de una serie de bloques que almacenan los datos del archivo, de esta forma si el archivo es pequeño solo se necesitarán estos punteros. El resto se referenciarán a bloques indirectos, distinguiendo entre uno, dos o tres niveles. Esto permite que podamos tener archivos de gran tamaño.

Prestaciones

Asignación contigua: Sólo un acceso para poder extraer un bloque de disco, podemos mantener fácilmente la dirección de acceso en memoria y calcular la dirección en disco.

Asignación enlazada: Mantenemos en memoria la dirección del siguiente bloque y lo leemos directamente. Acceso secuencial.

Asignación indexada: Más compleja. Si el bloque índice está en memoria, puede realizarse el acceso directo. Las prestaciones de este tipo de asignación dependerá de la estructura índice, tamaño del archivo y la posición del bloque deseado.

Combinación de contigua (acceso directo o secuencial) y enlazada (acceso secuencial).

Combinación de contigua (archivos menos tamaño) e indexada (archivos de gran tamaño).

Gestión de espacio libre

El sistema controla el espacio libre en disco mediante la **lista de espacio libre**. Cuando creamos un archivo, se explora la lista hasta encontrar el sitio adecuado para el nuevo archivo, a continuación, el espacio ocupado se elimina de la lista, se volverá a añadir cuando lo eliminemos.

La FAT incorpora un control de bloques libres, por tanto, no hace falta ningún método separado.

La lista de espacio libre tiene varias implementaciones:

1. **Mapa o Vector de bits**:

Cada bloque se representa con un bit (0 libre, 1 ocupado).

Este método es simple y eficiente, permite localizar los bloque libres consecutivos. Algunas máquinas tienen instruciones específicas.

Es ineficiente si no se mantiene en memoria principal y esto es posible si los archivos son de tamaño pequeño.

1. **Lista enlazada**:

Enlaza todos los bloques libres de disco, guarda un puntero al primer bloque en lugar concreto en disco y en memoria caché.

No derrocha espacio.

Relativamente ineficiente, para recorrer la lista debemos leer cada bloque. Pérdida de tiempo de E/S. Pocas veces tenemos que recorrer la lista.

1. **Lista enlazada con agrupación**:

Cada bloque de la lista almacena *n-1* direcciones de bloques libres. Se

pueden encontrar fácilmente gran número de direcciones libres.

1. **Cuenta**:

Generalmente se asignan y liberan simultáneamente varios bloque contiguos. Así, sólo tenemos que mantener la dirección del bloque libre y el número de bloques de disco libres, que será lo que se encuentra en cada entrada de la lista.

Implementación de directorios

Algoritmos de asignación de directorios

- **Lista lineal** \

```
Lista lineal de nombres de archivos con punteros a bloque
```

- **Tabla Hash** \

```
Inserción y borrado bastante sencillas. Es posible la apa
```

Contenido de una entrada de directorio

- Nombre de archivo + Atributos + Dirección de los bloques de datos

  * Nombre de archivo: Nombre con la extensión del archivo

  * Atributo: Espacio reservado, tiempos (acceso, modificación), tamaño

  * Dirección de los bloques: Número del primer bloque

- Nombre de Archivo + Puntero a una estructura de datos (Información relativa al archivo)\

Cuando abrimos un archivo:

- El SO busca en su directorio la entrada correspondiente
- Extrae sus atributos y la localización de sus bloques de datos y los coloca en un tabla en memoria principal
- Cualquier referencia posterior usa la información de dicha tabla

Implementación de archivos compartidos (Enlaces):

- Enlaces simbólicos:

  * Se crea una nueva entrada en el directorio, se indica que es de tipo enlace y se almacena el camino de acceso absoluto o relativo del archivo al cual se va a enlazar.

  * Se puede usar en entornos distribuidos.

  * Gran número de accesos a disco.

- Enlaces absolutos:

  * Se crea una nueva entrada en el directorio y se copia la dirección de la estructura de datos con la información del archivo.

  * Problema al borrar los enlaces, se puede solucionar con el contador de enlaces.

Distribución de los sistemas de

archivos

Los sistemas de archivos se almacenan en discos que pueden dividirse en una o más particiones.

Existe un bloque de arranque para inicializar el sistema localizado por bootstrap, que busca las particiones activas y las carga en memoria.

Formateo del disco:

- Físico: Pone los sectores (cabecera y código de corrección de errores) por pista.
- Lógico: Escribe la información que el SO necesita para conocer y mantener los contenidos del disco.

Son necesarios algunos métodos para detectar y manejar bloques dañados.

4.3. Implementación de la Gestión de Archivos en Linux

i-nodo: Representación interna de un archivo, es único para cada archivo aunque puede tener distintos nombres. Este i-nodo se asigna cuando se crea el archivo y se lleva a memoria hasta que se cierre.

- *Identificador del propietario*: UID, GID
- *Tipo de archivo*: regular, directorio, dispositivo, cauce, link, 0 si está libre
- *Permisos de acceso*
- *Tiempos de acceso*: última modificación, último acceso y última vez

que se modificó el i-nodo
- *Contador de enlaces*
- *Tabla de contenidos* para las direcciones de los datos en disco del archivo.
- *Tamaño*

Implementación de Sistemas de Archivos

Información acerca de cómo iniciar un sistema operativo:

- **Bloque de control de arranque**: información que necesita el sistema para iniciar al sistema operativo a partir del directorio
- **Bloque de control de directorio**: detalles acerca de la partición (número de bloques, tamaño de los bloques, bloques libres...)
- **Estructura de directorios** en la que se incluye el nombre del archivo y el inodo asociado

Los datos se cargan en el momento de montaje del directorio y se descartan cuando el dispositivo se desmonta. Las estructuras existentes pueden ser:

- Tabla de montaje en memoria que contiene información del directorio montado
- En caché se guarda información sobre los directorios que se han accedido recientemente
- Tabla global de archivos abiertos, contiene una copia del archivo de control de bloque de cada archivo abierto

- Tabla de archivos abiertos de cada proceso, contiene un puntero a la entrada apropiada de la entrada global de archivos abiertos
- Bloque de control de archivos (FCB), contiene los permisos del archivo, fechas del archivo, propietario del archivo, tamaño del archivo, bloques de datos del archivo o punteros a los bloques de datos

Montaje y Desmontaje de SA

Las particiones sin formato no contienen ningún sistema de archivos. Se almacena información como los mapas de bits.

La información de arranque se almacena en una partición independiente, no puede interpretar ningún formato puesto que el sistema todavía no ha arrancado.

Realizaremos un arranque dual si tenemos una partición con distintos sistemas operativos.

La partición raíz contiene el kernel y los archivos que se montan en el momento de arranque.

Tabla de montaje: Cada entrada por cada sistema de archivo montado:

- Número de dispositivo que identifica el SA montado
- Puntero a un buffer que contiene una copia del superbloque
- Puntero al i-nodo raíz del SA montado
- Puntero al i-nodo del directorio punto de montaje

Tipos de Sistemas de Archivos

- **SA Basados en Disco**: almacenan archivos en medios no volátiles
- **SA Virtuales**: generados por el kernel, constituyen una forma simple para permitir la comunicación entre los programas y los usuarios. No requieren espacio de almacenamiento en ningún dispositivo hardware, la información está en memoria principal
- **SA de Red**: acceden a los datos a través de la red

Modelo de archivo común

Para un programa de usuario, un archivo se identifica con un descriptor de archivo, número que se usa como índice en la tabla de descriptores que identifica al archivo en las operaciones. Este descriptor lo asigna el kernel cuando se abre dicho archivo y solo es válido dentro del proceso.

Dos procesos pueden usar el mismo descriptor, pero no apuntará al mismo archivo.

Descripción de un Grupo de Bloques

- **Superbloque**: estructura central para almacenar meta-información del SA
- **Descriptores de grupo**: contienen información que refleja el estado de los grupos de bloques individuales del SA (bloques e i-nodos libres)
- **Mapa de bits de bloques de datos e inodos**: contienen un bit por bloque de datos y por i-nodo respectivamente para indicar si están libres o no
- **Tabla de i-nodos**: contienen todos los i-nodos del grupo de bloques. Cada i-nodo mantiene los metadatos asociados con un archivos o directorio del SA

Información sobres los bloques de un archivo

Linux utiliza un método de asignación de bloques no contiguo y cada bloque de un SA se identifica por un número.

El i-nodo almacena direcciones directas al bloque de datos, además de uno, dos o tres niveles de indexación.