¿Recuperación De Archivos A Través Del Virtual File System De Linux?

Víctor Mejía - Diego Guerrero
Directora: Ingeniera Claudia P. Santiago

Resumen: *Este artículo explica en primera instancia el por que el Virtual File System de Linux se puede llegar a considerar una opción para hacer la recuperación de archivos de forma genérica; y en segundo lugar se darán razones de peso que permitan responder la pregunta ¿Es posible hacer la recuperación de archivos de forma genérica a través del Virtual File System?.*

Abstract: *This article primarily explains the reason why Virtual File System of Linux can be consider as an option to recuperate files in a general form. Consequently substantial reasons will be given to answer the question, is it possible to recuperate files in a general form thru Virtual File System?; Therefore, the principal VFS data structures are analyzed, and at the end of the article the future work is enunciated and the conclusions about the analysis are written.*

I. INTRODUCCIÓN

Teniendo en cuenta el gran valor que la información tiene hoy en día para las diferentes entidades y que por ésta se ha convertido en uno de los blancos mas apetecidos del crimen (crimen cibernético), el cual se manifiesta de diferentes formas, una de ellas es borrando información, que por un lado puede ser valiosa para alguien o bien, contiene evidencia incriminatoria sobre algún evento delictivo; violando de ésta forma el principio de disponibilidad de la información en la seguridad informática.

Dada la gravedad de esta situación se hace importante contar con herramientas que realicen la recuperación de la información después de un evento de borrado de la misma sobre un medio de almacenamiento, ya que en la mayoría de los casos resulta muy costoso e impracticable tener una seguridad que garantice que el borrado de información es imposible sin una previa autorización.

Actualmente, Linux actualmente cuenta con un buen número de herramientas para la recuperación de archivos basándose en el sistema de archivos, algunas de ellas de licencia GPL; sin embargo, hasta donde se conoce, ninguna de ellas hace la recuperación sobre cualquier sistema de archivos manejado por Linux. Por esta razón puede resultar de interés construir una herramienta que funcione de manera genérica sobre cualquier sistema de archivos; tarea complicada o incluso imposible con lo que en la actualidad ofrece Linux en el manejo de sistemas de archivos. Con este artículo se pretende dar a conocer un componente de software con el que cuenta el kernel de Linux, que seguramente será un de las posibilidades que se tendrán en cuenta en el momento de tratar de encontrar la forma de construir la herramienta descrita anteriormente, nos estamos refiriendo al Sistema Virtual de Archivos de Linux (*Virtual File System*

o VFS), enfocándonos en las características que nos puedan ayudar a la hora de implementar dicha herramienta.

II. PROBLEMÁTICA

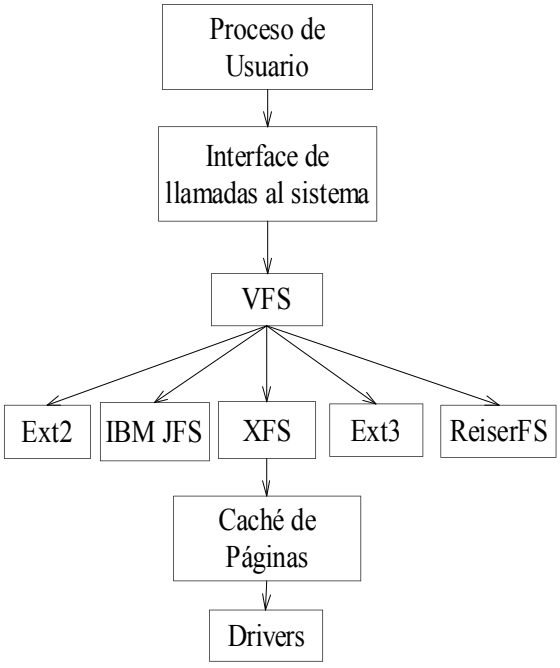
Si estamos en la tarea de construir una herramienta para recuperación de archivos, y conocemos de una forma básica el VFS, podemos pensar intuitivamente que la solución, para el problema de dar la portabilidad a la herramienta de forma genérica sobre los diferentes sistemas de archivos manejados por Linux, es por medio del VFS.

En este artículo se pretende dar respuesta a la pregunta de si es o no viable construir una herramienta que recupere archivos sobre cualquier sistema de archivos soportado por Linux (Ext2, Ext3, ReiserFS, XFS, etc.) y que se base en sus sistema virtual de archivos (VFS).

Es claro que, de darse una respuesta negativa, es importante descartar posibilidades de trabajo en el problema de construir una herramienta de recuperación de archivos, que funcione de manera genérica sobre cualquier sistema de archivos manejado por Linux, ya que esto ahorrará muchas horas de trabajo tratando de entender algo que definitivamente no nos llevará a cumplir con nuestro objetivo; y por otro lado, de obtener una respuesta positiva solo queda profundizar sobre el tema y empezar a desarrollar la herramienta de recuperación de archivos.

III. MARCO TEÓRICO

Para el manejo de archivos Linux provee una herramienta muy potente, la cual fue diseñada con el fin de dar soporte a varios sistemas de archivos. Esta herramienta es la que se conoce como Sistema de Archivos Virtual (VFS Virtual File System). El VFS presenta una interfaz para que los procesos puedan trabajar con archivos; en este se definen abstracciones sobre las características y funciones de los sistemas de archivos particulares e implementa las funciones comunes. Se puede ver al VFS como un intermediario entre los procesos de los usuarios y los sistemas de archivos reales como los son Ex2, Ex3, XFS, entre otros.
La siguiente figura describe a grandes rasgos este comportamiento:



En términos de la programación orientada a objetos, se puede decir que el VFS es una clase abstracta que implementa funciones generales, delega funciones específicas a los sistemas de archivos particulares y provee una serie de interfaces para que los procesos puedan acceder a dichas funciones.

El sistema virtual de archivos de Linux es el software encargado de manejar todas las llamadas al sistema relacionadas con el sistema de archivos.

Su principal fortaleza es proveer un grupo de interfaces para una gran cantidad de tipos de sistemas de archivos.

Los sistemas de archivos soportados por el sistema virtual de archivos se pueden clasificar en tres clases:

- *Disk-based filesystems*
- *Network filesystems*
- *Special filesystems*

Para el tema que nos ocupa en este documento solo hablaremos del primero de la clase *Disk-based filesystems*, la cual se encarga de manejar el espacio de memoria disponible en una partición de un disco o en algún medio de almacenamiento. Normalmente realiza el almacenamiento por medio de bloques en el dispositivo de hardware.

IV. DESARROLLO DE LA PROPUESTA

El VFS para su funcionamiento usa tres objetos o estructuras de datos principales, las cuales a su vez usan otras estructuras de datos que se clasifican por su objetivo; entonces, de estas últimas estructuras tenemos:

- Estructuras de datos que contienen las operaciones que se pueden realizar sobre la estructura principal que los contiene.
- Estructuras que contienen información importante, para la correcta administración de los archivos y para mantener la compatibilidad con los sistemas de archivos concretos (Ext3, Ext2, JFS, etc.).
- Por último tenemos las estructuras que tienen como objetivo mejorar el rendimiento del sistema operativo. Básicamente se trata de cachés, que ayudan a evitar el exceso de transacciones con operaciones de E/S al dispositivo de almacenamiento.

Dado que el rendimiento no es el tema de interés para este artículo, se omitirá hablar del manejo de cachés del VFS y nos centraremos, para cada una de las tres estructuras principales del VFS, en las estructuras de datos que ellos manejan y cuyos objetivos caen en los dos primeros descritos anteriormente.

Comenzaremos hablando del objeto o estructura *superbloque*. Esta estructura posee la información concerniente a cada sistema de archivos particular registrado en el VFS; dicha información incluye:

- Un apuntador a una lista de superbloques, en donde cada superbloque de esta lista representa un sistema de archivos específico.
- El tamaño del bloque en bytes y en bits que usa el sistema de archivos específico.
- El tipo del sistema de archivos (Disk-based, Network o Special).
- Banderas y Semáforos de control.
- Una lista de los inodos modificados en memoria que no han sido actualizados en disco.
- Una lista de los objetos *File* que se están usando en el momento.
- Un apuntador al bloque inicial o el punto de montaje del sistema de archivos.
- El dispositivo donde fue montado el sistema de archivo
- Un apuntador a una estructura que tiene las operaciones que se pueden realizar sobre el superbloque
- Una unión (en el sentido del lenguaje C), donde cada sistema

de archivos almacena una información específica de él.

De toda esta información se podría considerar de interés para el tema que nos ocupa el tamaño de los bloques, que sería útil para nuestro fin si tuviéramos la certeza que los bloques usados por todos los sistemas de archivos tienen la misma estructura, algo no ocurre.

En cuanto a el conjunto de métodos definidos para el superbloque se tienen de dos tipos; el primero maneja métodos que operan sobre los inodos tales como lectura y escritura, que variarán dependiendo del tipo de inodo (si es inodo raíz o un inodo común), también se puede revisar el estado de un inodo (si fue modificado o no) y varios tipos de borrado del inodo a nivel de memoria y a nivel de disco, en este último se pierde referencia a toda la información del archivo que representaba el inodo borrado. El segundo tipo de métodos opera sobre el superbloque y permiten lectura, escritura, revisión del estado del superbloque y revisión de las opciones definidas por el sistema de archivos que representa el superbloque.

Ahora se hablará de la estructura *inodo*, la cual tiene una relación muy estrecha con los archivos, ya que para cada archivo existe un único inodo que lo representa en el sistema de archivos. El inodo maneja información muy importante sobre cada archivo, la cual es necesaria para que el sistema de archivos pueda manejar los archivos, esta información se resume en la siguiente lista.

- Número de inodo (los inodos se identifican por este número).
- Un contador del uso del inodo.
- Un identificador del dispositivo donde opera.
- Identificador del propietario del archivo y del grupo al que pertenece.
- Tamaño del archivo en bytes.
- Fechas del último acceso, de la última modificación y del último cambio en el inodo.
- Tamaño del bloque usado por el sistema de archivos.
- Un apuntador a una estructura que tiene las operaciones que se pueden realizar por defecto sobre una estructura *File*.
- Un apuntador a una estructura que tiene las operaciones que se pueden realizar sobre una estructura inodo.
- Referencias a bloques del dispositivo donde se encuentra el archivo que representa.
- Apuntador al superbloque del sistema de archivos al que pertenece el inodo.
- Banderas y semáforos de control.
- El inodo también tiene una unión que tiene información específica de cada sistema de archivo.

Como nos podemos dar cuenta, toda esta información es fundamental para que el sistema de archivos pueda, no solo ubicar el archivo en el dispositivo de almacenamiento, si no que también permite saber el propietario, para poder hacer control de permisos sobre el archivo, entre otras cosas; todo esto lo hace a través de las operaciones definidas para el inodo. Sin embargo el VFS solo mantiene esta información para archivos que no han sido borrados, y depende de cada sistema de archivos el mantener o no las referencias a bloques de archivos borrados, aunque sea por un tiempo mientras se vuelven a asignar los bloques.

La última estructura u objeto es el *File*, el cual almacena información sobre la

interacción que existe entre un archivo abierto y un proceso, y se crea en respuesta a una llamada al sistema *open()* y se destruye en respuesta al llamado *close()*. Tal información existe únicamente en la memoria del kernel durante el periodo que cada proceso accede el archivo, es decir no es persistente luego no vale la pena entrar en detalles sobre esta estructura.

Toda la información y operaciones que se han descrito para el superbloque y el inodo pueden ayudar a la hora de recuperar un archivo, sin embargo hace falta, por lo menos, conocer información sobre que bloques hay disponibles y conocer la estructura o el formato que se maneja para cada bloque, ya que es necesario diferenciar e interpretar la información de control almacenada en el bloque y de este modo poder reconstruir la información total o parcial del archivo.

Dado que cada sistema de archivos puede manejar un formato de bloque y una información de control, que no es necesariamente es común a los otros sistemas de archivos, debido a que cada sistema de archivos pudo ser diseñado con diferentes propósitos y por diferentes personas, las cuales le dieron una estructura que a nivel general es muy similar a todos los sistemas de archivos que es lo que el VFS cubre, pero las características particulares las que a la hora, de hacer la recuperación de archivos marcan la diferencia.

Vale la pena aclarar que lo único que se ha afirmado aquí es que no es posible, que usando únicamente las estructuras del VFS, hacer la recuperación de archivos, lo que no quiere decir que no se pueda usarlas, ya que estas tienen información que puede ser de interés a la hora de hacer la recuperación.

V. CONCLUSIONES

- El VFS es una capa de software que abstrae de los sistemas de archivos, información y funcionalidad común, sin embargo hay funcionalidad crítica que es diferente para cada sistema de archivos particular
- Para poder hacer la recuperación, es necesario conocer, cómo es que se guardan, y mas importante aún cómo se borran los archivos, y la forma de realizar estas tareas es particular para cada sistema de archivos.
- Es necesario para recuperar un archivo, leer entradas de los directorios, analizar los inodos correspondientes a los archivos **borrados** (que no son manejados por el VFS), y seguir los bloques referenciados por dicho inodo; además se debe contar con la suerte que el sistema de archivos particular, no haya asignado tales bloques (esto depende del sistema de archivos y el tiempo transcurrido desde que se borró el archivo).
- El VFS maneja información que puede ayudar a orientar una eventual búsqueda de un archivo borrado (entre mas información se tenga, se puede mejorar la heurística de búsqueda haciendo que ésta sea más eficiente).

BIBLIOGRAFÍA

- [ULK2] Understanding The Linux Kernel 2nd / Daniel P. Bovet, Marco Cesati. Ed. O'Reilly Diciembre 2002
- [SOS5] Sistemas Operativos Quinta Edición / Willinam Stallings. Ed. Pearson. 2005

- El Linux Virtual File System, Juan Antonio Martínez
  En: http://es.tldp.org/Articulos-periodisticos/jantonio/ el 25/Agosto/2006

Víctor Andrés Mejía, Estudiante de décimo semestre de Ingeniería de Sistemas de la Escuela Colombiana de Ingeniería, participante a lo largo de la carrera en el desarrollo de diversos proyectos entre los más destacados están: SME "Desarrollo de un Sistema Manejador de Encuestas para Mercadatos" donde se encargó del diseño y construcción de dos casos de uso.
FWJDC "Desarrollo de un Framework de Juegos Discretos con Contrincante", desarrollado en el Segundo semestre del 2005, es otro proyecto que tuvo un gran éxito, en el cual desempeñó el rol de Líder de Desarrollo.
También cuenta con experiencia trabajando con las soluciones JES Directory Server (implementación de LDAP de Sun) y JES Messaging Server.
vamsp7@yahoo.com

Diego Andrés Guerrero Aguirre, Estudiante de décimo semestre de Ingeniería de Sistemas de la Escuela Colombiana de Ingeniería, constante investigador de las diferentes ramas de la computación, y tecnologías, certificado en "Computer Fundamentals (Win XP)" en Marzo de 2003, participante de varios proyectos de software libre desarrollados como estudiante de La Escuela Colombiana de Ingeniería, tales como lo son la construcción de un portal para publicación de proyectos de Software Libre, construido para Noviembre de 2005, desarrollador de SSConstruir, sistema manejador de facturación para una organización no gubernamental llamada Corporación Construir (Enero – Abril 2006), participante en el diseño y desarrollo de SIPLA 2, software para el manejo de lavado de activos, construido por la empresa IBISCOM (Marzo – Abril 2006).
diego.guerrero.a@gmail.com.