INSTITUTO POLITÉCNICO NACIONAL ESCUELA SUPERIOR DE CÓMPUTO

INGENIERÍA EN SISTEMAS COMPUTACIONALES

SISTEMAS OPERATIVOS

David Díaz Araujo

Practica 12.

Control de acceso a los recursos

- Alumnos:
  1) Arcos Hermida Aldo Alejandro (N° lista 5)
  2) Chávez Becerra Bianca Cristina (N° lista 9)
  3) Islas Osorio Enrique (N° lista 20)
  4) Juárez Cabrera Jessica (N° lista 21)
  5) Palmerin García Diego (N° lista 28)
- Grupo: 4CM1
- Materia: Sistemas operativos
- Profesor: Araujo Díaz David

Índice

Objetivo

Describir como se administran los recursos en los sistemas operativos LINUX/UNIX.

Descripción

Historia

A lo largo de la historia, cada uno de los sistemas operativos que conocemos, ha tenido su propio “Sistemas de archivos”. Así, podemos encontrar que Microsoft trabaja con dos tipos de sistemas de archivos bien identificados: FAT (que tenía varias versiones, y que era utilizado para Microsoft Windows 95, Windows 98, y Windows XP) y NTFS (propio de Microsoft Windows NT, Windows XP, Windows Vista o el actual Windows 7). Son muchas las diferencias entre FAT y NTFS, pero las más importantes están orientadas a la posibilidad de contar con dispositivos de almacenamiento más grandes (FAT estaba limitado en tamaño de particiones) y más seguridad en el acceso a los ficheros del sistema.

Por el contrario, GNU/Linux comenzó su andadura con su sistema de archivos “Ext2”, pero éste fue sustituido por nuevas versiones que si tenían la capacidad de trabajar con grandes volúmenes de información (Terabytes) y de poder restituir rápidamente (gracias al “Journaling”) el sistema ante un fallo importante en el sistema de archivos. Posteriormente aparecieron otros sistemas de archivos nuevos y mejoras sobre el modelo existente. Veámoslo con más detalle:

Ext2	**(Sistema de archivos Extendido, versión 2)**el primer sistema de archivos utilizado por GNU/Linux fue creado por Remy Card (no es del todo cierto, antes existió "Ext", utilizado con "Minix", pero debido a sus limitaciones y usos no suele contar como un sistema de archivos válido para las distribuciones GNU/Linux que conocemos hoy en día). Como todos los sistemas de archivos de Linux, éste también es asíncrono, es decir, no escribe inmediatamente los metadatos en el dispositivo de almacenamiento, sino que lo hace de manera periódica. Con ello consigue aprovechar los tiempos muertos de la CPU y consecuentemente, el rendimiento general del equipo. Pese a ser el primero, ya dispone de mecanismos que permiten la recuperación de la información en caso de fallo (detectando particiones desmontadas erróneamente).
Ext3	**(Sistema de archivos Extendido, versión 3)**es compatible con Ext2 (la única diferencia con éste es que posee un fichero adicional de registro para implementar "journaling"). De hecho, el objetivo de Ext3 era mejorar Ext2, pero manteniendo la compatibilidad con éste. Entre las principales diferencias cabe destacar que Ext3 mantiene la consistencia no solo de los metadatos (como ya hace Ext2) sino también de los propios datos. Por supuesto, la seguridad de poder recuperar los datos de nuestro sistema tiene un coste, y es que tendremos menos rendimiento y más consumo de espacio en disco.
Ext4	**(Sistema de archivos Extendido, versión 4)**mantiene la compatibilidad con sus antecesores, posee "journaling", reduce considerablemente la fragmentación de archivos (mejorando con ello el rendimiento), permite dispositivos de almacenamiento de más capacidad.
ReiserFS	desarrollado por la empresa Namesys, fue el primer sistema de archivos con "journal" incluido en un núcleo estándar de Linux. Pero además, implementa otra serie de ventajas no disponibles en otros sistema de archivos, como puede ser la repartición de sistemas de ficheros montados, o un esquema para reducir la fragmentación. La versión más reciente de este sistema de archivos se denomina "Reiser4", y además de las características antes indicadas, posee mecanismos que le permiten trabajar con cientos de miles de archivos, y una estructura de archivos optimizada.
XFS	creado por Silicon Graphics Inc., se trata de un sistema de archivos con "journaling" que inicialmente funcionaba sobre la implementación IRIX de UNIX, pero que posteriormente se liberó como código abierto. Destaca por su alta escalabilidad y fiabilidad (admite redireccionamiento de 64 bits, implementación paralelizada), y sobre todo porque es capaz de trabajar con archivos muy grandes.
JFS	desarrollado por IBM, inicialmente fue creado para servidores de alto rendimiento y equipos de altas prestaciones. Posee un eficiente "journaling" que le permite trabajar cómodamente con archivos de gran tamaño como con otros más pequeños. Las particiones JFS pueden ser dinámicamente redimensionadas (como ya hacía RaiserFS), pero no pueden ser comprimidas (algo que si hacen RaiserFS y XFS).

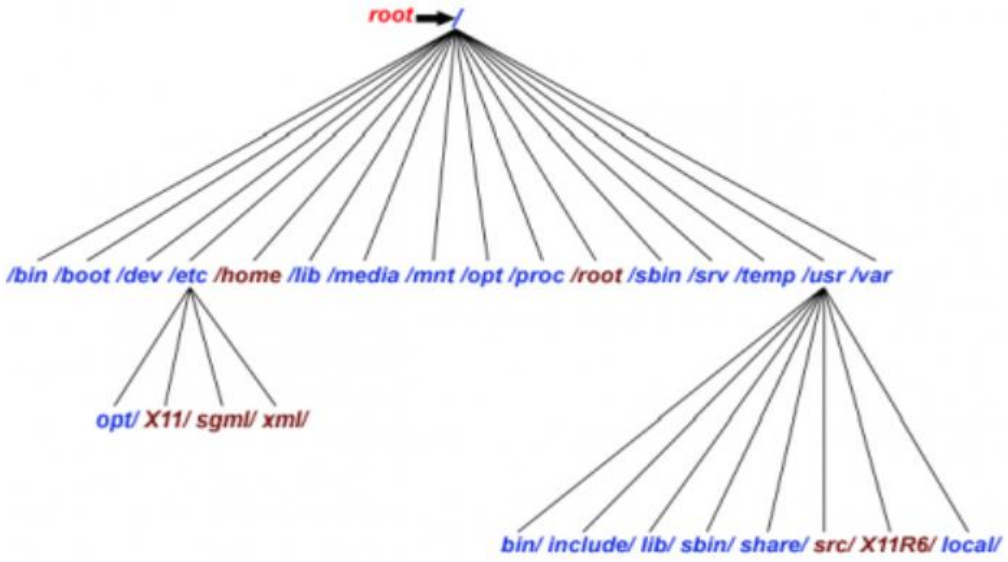
Journaling

Seguro que no le ha pasado desapercibido el término “Journaling” o “journal” a lo largo de los párrafos anteriores, y es que se trata de la principal característica que identifica los sistemas operativos modernos. El “Journaling” (también conocido como “registro por diario”) es un mecanismo que almacenará las transacciones (operaciones de lectura y escritura de archivos) que se realizan en el sistema, y que permitirá la recuperación de los datos en caso de fallo grave.

Su funcionamiento, a grandes rasgos, sería el siguiente: cuando se desea guardar un archivo en el sistema de ficheros, se anotará esta acción en el “journal” (en ocasiones también se le denomina “bitácora”) pasando a continuación a realizar realmente la operación de escritura en el disco duro del equipo. Si la operación de escritura en disco duro finaliza con éxito entonces se elimina esa operación del “journal” (“diario”). Pero, si antes de finalizar la operación de escritura en disco se produjese un fallo (por ejemplo, un corte eléctrico) entonces el “journal”, que aún mantiene la información correspondiente al fichero, podría recuperarlo para el sistema rápidamente en el siguiente inicio.

Estructura de Directorios en Linux

El estándar utilizado por GNU/Linux para organizar la información se denomina FHS (Filesystem Hierarchy Standard), y éste sistema se encarga de organizar la información de forma jerárquica.



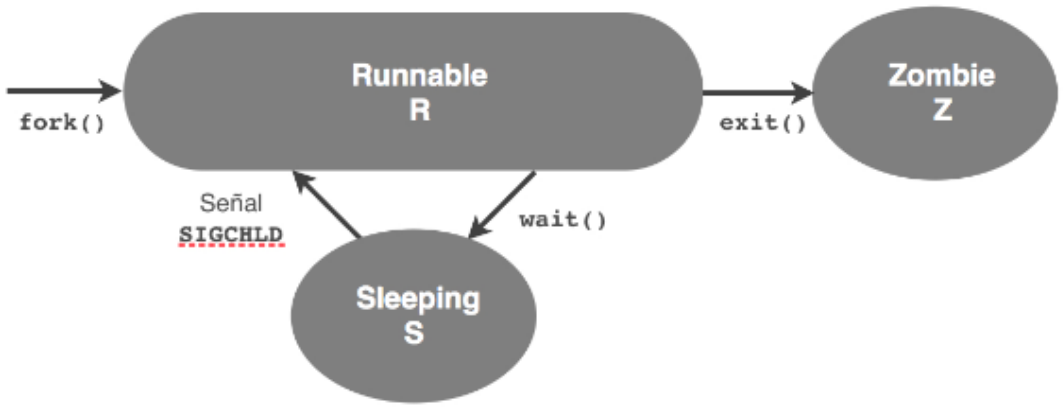
Partiendo de un “raiz” encontraremos los siguientes directorios:

/bin	almacena las aplicaciones (comandos) básicas del sistema.
/boot	aquí se encontrarán los archivos necesarios para el inicio del sistema, así como los correspondientes al cargador de arranque.
/dev	cada uno de los archivos representa a un dispositivo del sistema.
/etc	es el directorio donde se encontrarán la mayoría de los archivos de configuración del sistema y de otras aplicaciones importantes.
/home	donde se encontrarán los directorios personales de los usuarios del sistema.
/lib	bibliotecas compartidas necesarias para la ejecución del sistema.
/mnt	se trata del directorio en el que se solían ‘montar’ los distintos dispositivos de almacenamiento (discos duros externos, pen-drive), pero que ahora ha quedado obsoleto porque se utiliza el nuevo directorio “/media” para dicha función.
/proc	mantiene ficheros que almacenan el estado (procesos, dispositivos) del sistema.
/root	es el directorio personal del administrador del sistema.
/sbin	comandos de administración del sistema.
/tmp	carpeta donde el sistema almacena información temporal.
/usr	ubicación que normalmente se dedica para instalar las aplicaciones de usuario.
/var	su contenido no se explica brevemente, ya que en él podremos encontrar los archivos de registro del sistema, archivos temporales del servicio de correo, o el directorio de trabajo del servidor de páginas web.

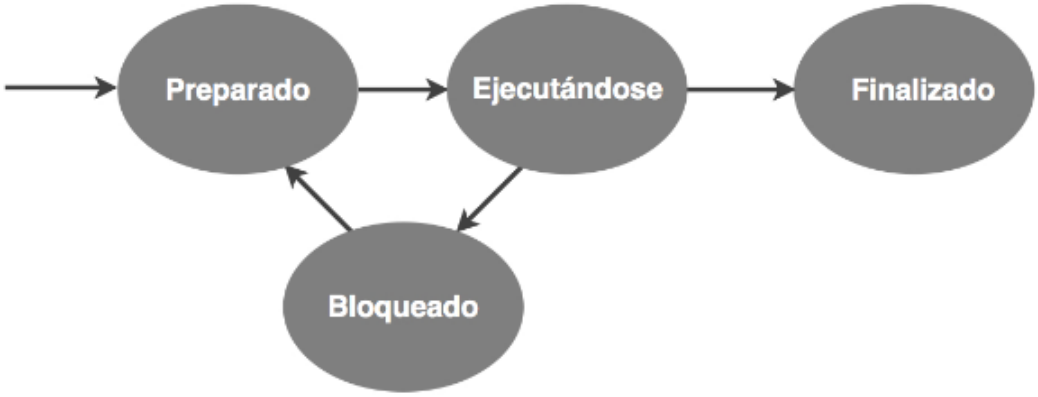
Gestión de procesos

Ya conocemos el concepto de proceso UNIX como entidad que representa un flujo de ejecución con un contexto asociado. Los procesos son las entidades clave de los sistemas operativos multiprogramados como UNIX/Linux y son la base para un tipo de programación concurrente.

Un proceso consume recursos, fundamentalmente tiempo de CPU, además de memoria para almacenar el contexto y de otros recursos relacionados con la entrada/salida. Típicamente, un proceso requiere la CPU durante un periodo de tiempo, realiza alguna operación de entrada/salida, y vuelve a requerir la CPU, repitiéndose este ciclo hasta la finalización del programa. Durante su ejecución, el proceso pasa por diversos estados, que representábamos para un ejemplo particular sobre Linux mediante un grafo de transiciones de estado:

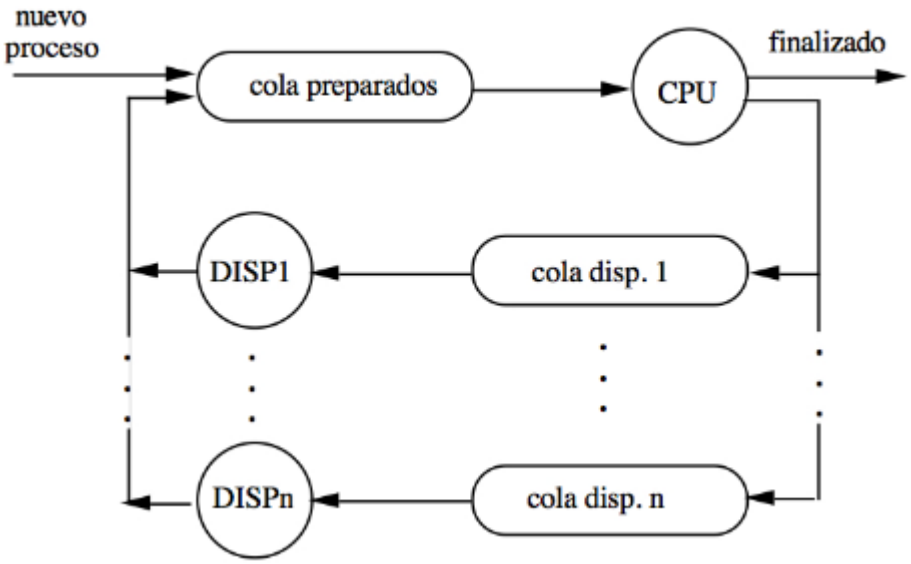


Por generalizar, llamaremos CPU al recurso de procesamiento, ya sea un procesador o un core. Esta simplificación nos mantendrá a salvo de los detalles farragosos al tiempo que nos permitirá comprender la mayor parte de las implicaciones sobre el rendimiento. El número absoluto de CPUs no es relevante salvo que nos dediquemos a la computación de altas prestaciones. Más interesante para la mayoría de las aplicaciones resulta el tiempo que los procesos disfrutan de CPU frente al que están esperando, de la misma forma que cuando salimos de un supermercado no nos fijamos tanto en el número de cajas como en la longitud de las colas. Como punto de partida para este propósito, resulta conveniente definir un diagrama de transición de estados que distinga la espera por CPU del uso de la CPU por el proceso. De esta forma, podemos representar el siguiente grafo de transición de estados general:



Representación de los procesos

A partir del grafo resulta sencillo imaginar el conjunto de procesos en el sistema operativo como un sistema de colas con procesos esperando por recursos. El estado Preparado representaría una cola de procesos esperando CPU. Recuerda que el estado Bloqueado es múltiple, por lo que se representaría mediante un conjunto de colas, una por estado de bloqueo (por ejemplo, entrada de teclado, señales…). En general, si consideramos que un proceso se bloquea para usar un dispositivo de entrada/salida (aunque no siempre es así), podríamos ilustrar el funcionamiento del sistema operativo de la siguiente forma:



Esto nos permite hacernos una idea bastante intuitiva de cómo se gestionan los procesos en un sistema operativo. Faltan muchos detalles, como definir la disciplina en las colas (la de la caja del supermercado es FIFO, pero no tiene por qué ser el caso aquí, como veremos), o cómo se representa un proceso.

Tipos de usuarios

Los usuarios en Unix/Linux se identifican por un número único de usuario, User ID, UID. Y pertenecen a un grupo principal de usuario, identificado también por un número único de grupo, Group ID, GID. El usuario puede pertenecer a más grupos además del principal.

Aunque sujeto a cierta polémica, es posible identificar tres tipos de usuarios en Linux:

Usuario root
También llamado superusuario o administrador.
Su UID (User ID) es 0 (cero).
Es la única cuenta de usuario con privilegios sobre todo el sistema.
Acceso total a todos los archivos y directorios con independencia de propietarios y permisos.
Controla la administración de cuentas de usuarios.
Ejecuta tareas de mantenimiento del sistema.
Puede detener el sistema.
Instala software en el sistema.
Puede modificar o reconfigurar el kernel, controladores, etc.
Usuarios especiales
Ejemplos: bin, daemon, adm, lp, sync, shutdown, mail, operator, squid, apache, etc.
Se les llama también cuentas del sistema.
No tiene todos los privilegios del usuario root, pero dependiendo de la cuenta asumen distintos privilegios de root.
Lo anterior para proteger al sistema de posibles formas de vulnerar la seguridad.
No tienen contraseñas pues son cuentas que no están diseñadas para iniciar sesiones con ellas.
También se les conoce como cuentas de "no inicio de sesión" (nologin).
Se crean (generalmente) automáticamente al momento de la instalación de Linux o de la aplicación.
Generalmente se les asigna un UID entre 1 y 100 (definifo en /etc/login.defs)
Usuarios normales
Se usan para usuarios individuales.
Cada usuario dispone de un directorio de trabajo, ubicado generalmente en /home.
Cada usuario puede personalizar su entorno de trabajo.
Tienen solo privilegios completos en su directorio de trabajo o HOME.
Por seguridad, es siempre mejor trabajar como un usuario normal en vez del usuario root, y cuando se requiera hacer uso de comandos solo de root, utilizar el comando su.
En las distros actuales de Linux se les asigna generalmente un UID superior a 500.

Conclusiones

Linux es un sistema multiusuario, por lo tanto, la tarea de añadir, modificar, eliminar y en general administrar usuarios se convierte en algo no solo rutinario, sino importante, además de ser un elemento de seguridad que mal administrado o tomado a la ligera, puede convertirse en un enorme hoyo de seguridad. Por otra parte el sistema de ficheros hace referencia a la forma en la que la información se organizará en el disco duro

Referencias

Digital Learning. (2014, 27 junio). Administración Linux: Sistemas de Archivos | Digital Learning. Digital Learning | Formación online en Nuevas Tecnologías. Recuperado 5 de junio de 2022, de https://www.digitallearning.es/blog/administracion-linux-sistemas-de-archivos/

Gestión de procesos — Programación concurrente en Linux. (s. f.). ocw. Recuperado 5 de junio de 2022, de https://ocw.ehu.eus/pluginfile.php/48902/mod_resource/content/13/html/Recursos/P03/Gestion_procesos.html

3.6 Administración de recursos - Ing Sistemas Computacionáles. (s. f.). google. Recuperado 5 de junio de 2022, de https://sites.google.com/site/jachsistemascomputacionales/36-administracin-de-recursos