Facultad de Contaduría y Administración

Sistema de Universidad Abierta

Apuntes
SISTEMAS OPERATIVOS MULTIUSUARIO

Profesor: L.A. Salvador Meza Badillo

Índice

I. DEFINICIÓN DE LOS CONCEPTOS FUNDAMENTALES

Definición de sistema operativo

Un sistema operativo es un tipo de software, es decir, forma parte de la parte lógica del sistema, que es, sobre todo, un software de sistema, ya que tiene la función de cooperar con el usuario para que este se comunique eficazmente con el hardware y el software de aplicación. En los últimos tiempos, aparte de este objetivo principal, se ha agregado otro secundario, como es el de apartar a los usuarios, en la medida de lo posible, de las instrucciones máquina del equipo.

Al utilizar un sistema informático se requieren de él recursos, como por ejemplo, procesos de trasvase de información, gestión de memorias, controladores, etc. Para obtener un sistema eficiente, es necesario que la gestión de esos recursos sea la óptima, por lo que se necesita de un programa como el sistema operativo. Esa eficiencia se pretende también para la gestión de los periféricos de entrada y de salida -impresoras, teclados, punteros,...-, y también para los dispositivos de entrada y de salida -discos, cintas,...-.

Además, el usuario de un sistema no se contentará sólo con usar éste pero , sino que querrá ejecutar otros programas de aplicación, que es lo que justifica, al fin y al cabo, el invento de los equipos informáticos. El sistema operativo gestiona, de la manera más eficiente y eficaz posible, las relaciones entre esos programas y todo el sistema. Es decir, es un controlador lógico de todo el proceso y flujo de información, así como de la ejecución de dichos programas.

Funciones de los sistemas operativos

Una primera aproximación a las tareas de un sistema operativo desde dos vertientes: asistencia a la programación y gestión del sistema. Desde el punto de vista de la primera función, las tareas de un sistema operativo se pueden resumir en:

- Asistencia en la compilación, depuración y linkado de programas
- Manejo y ejecución de programas, su carga en memoria, su ejecución y la finalización
- Gestión de la información, manejo de ficheros, modificación, lectura, grabación,...
- Controlar las operaciones de entrada y salida
- Detección de errores físicos o lógicos

Para la gestión eficiente del sistema informático el sistema operativo debe:

- Informar sobre el estado de uso del mismo, usuarios, tareas, tiempos, recursos,...
- Dar inicio a los procesos y mantenerlos hasta que acaben
- Interpretar los lenguajes de control
- Protección de la estructura de la información ante usos inadecuados: detectar y corregir errores, protección de datos, de programas
- Asignación y separación de recursos: CPU, memoria, dispositivos de entrada y salida,...
- Gestionar las interrupciones
- Interactuar con el usuario

1.- Aceptar todos los trabajos y conservarlos hasta su finalización.
2.- Interpretación de comandos: Interpreta los comandos que permiten al usuario comunicarse con el ordenador.
3.- Control de recursos: Coordina y manipula el hardware de la computadora, como la memoria, las impresoras, las unidades de disco, el teclado o el Mouse.
4.- Manejo de dispositivos de E/S: Organiza los archivos en diversos dispositivos de almacenamiento, como discos flexibles, discos duros, discos compactos o cintas magnéticas.
5.- Manejo de errores: Gestiona los errores de hardware y la pérdida de datos.

6.- Secuencia de tareas: El sistema operativo debe administrar la manera en que se reparten los procesos. Definir el orden. (Quien va primero y quien después).
7.- Protección: Evitar que las acciones de un usuario afecten el trabajo que esta realizando otro usuario.
8.- Multiacceso: Un usuario se puede conectar a otra máquina sin tener que estar cerca de ella.
9.- Contabilidad de recursos: establece el costo que se le cobra a un usuario por utilizar determinados recursos.

Características de los sistemas operativos.

En general, se puede decir que un Sistema Operativo tiene las siguientes características:

Conveniencia:
Un Sistema Operativo hace más conveniente el uso de una computadora.

Eficiencia:
Un Sistema Operativo permite que los recursos de la computadora se usen de la manera más eficiente posible.
Habilidad para evolucionar. Un Sistema Operativo deberá construirse de manera que permita el desarrollo, prueba o introducción efectiva de nuevas funciones del sistema sin interferir con el servicio.
Encargado de administrar el hardware. El Sistema Operativo se encarga de manejar de una mejor manera los recursos de la computadora en cuanto a hardware se refiere, esto es, asignar a cada proceso una parte del procesador para poder compartir los recursos.
Relacionar dispositivos (gestionar a través del kernel). El Sistema Operativo se debe encargar de comunicar a los dispositivos periféricos, cuando el usuario así lo requiera.
Organizar datos para acceso rápido y seguro.
Manejar las comunicaciones en red. El Sistema Operativo permite al usuario manejar con alta facilidad todo lo referente a la instalación y uso de las redes de computadoras.
Procesamiento por bytes de flujo a través del bus de datos.
Facilitar las entradas y salidas. Un Sistema Operativo debe hacerle fácil al usuario el acceso y manejo de los dispositivos de Entrada/Salida de la computadora.

Modalidades de trabajo de los sistemas operativos.

Sistemas operativos de tiempo compartido.

El tiempo compartido en ordenadores o computadoras consiste en el uso de un sistema por más de una persona al mismo tiempo. El tiempo compartido ejecuta programas separados de forma concurrente, intercambiando porciones de tiempo asignadas a cada programa (usuario). En este aspecto, es similar a la capacidad de multitareas que es común en la mayoría de los microordenadores o las microcomputadoras. Sin embargo el tiempo compartido se asocia generalmente con el acceso de varios usuarios a computadoras más grandes y a organizaciones de servicios, mientras que la multitarea relacionada con las microcomputadoras implica la realización de múltiples tareas por un solo usuario.

Los principales recursos del sistema, el procesador, la memoria, dispositivos de E/S, son continuamente utilizados entre los diversos usuarios, dando a cada usuario la ilusión de que tiene el sistema dedicado para sí mismo. Esto trae como consecuencia una gran carga de trabajo al Sistema Operativo, principalmente en la administración de memoria principal y secundaria.

Características de los Sistemas Operativos de tiempo compartido:

- Populares representantes de sistemas multiprogramados multiusuario, Ej.: sistemas de diseño asistido por computador, procesamiento de texto, etc.
- Dan la ilusión de que cada usuario tiene una máquina para sí.
- La mayoría utilizan algoritmo de reparto circular.

- Los programas se ejecutan con prioridad rotatoria que se incrementa con la espera y disminuye después de concedido el servicio.
- Evitan monopolización del sistema asignando tiempos de procesador (time slot).
- Gestión de memoria: proporciona protección a programas residentes.
- Gestión de archivo: debe proporcionar protección y control de acceso debido a que pueden existir múltiples usuarios accesando un mismo archivo.

Sistemas operativos de tiempo real.

Un sistema operativo en tiempo real procesa las instrucciones recibidas al instante, y una vez que han sido procesadas muestra el resultado. Este tipo tiene relación con los sistemas operativos monousuarios, ya que existe un solo operador y no necesita compartir el procesador entre varias solicitudes.
Su característica principal es dar respuestas rápidas; por ejemplo en un caso de peligro se necesitarían respuestas inmediatas para evitar una catástrofe.

Los Sistemas Operativos de tiempo real cuentan con las siguientes características:

- Se dan en entornos en donde deben ser aceptados y procesados gran cantidad de sucesos, la mayoría externos al sistema computacional, en breve tiempo o dentro de ciertos plazos.
- Se utilizan en control industrial, conmutación telefónica, control de vuelo, simulaciones en tiempo real., aplicaciones militares, etc.
- Su objetivo es proporcionar rápidos tiempos de respuesta.
- Procesa ráfagas de miles de interrupciones por segundo sin perder un solo suceso.
- Un proceso se activa tras ocurrencia de suceso, mediante interrupción.
- Un proceso de mayor prioridad expropia recursos.

Por tanto generalmente se utiliza planificación expropiativa basada en prioridades.
Gestión de memoria menos exigente que tiempo compartido, usualmente procesos son residentes permanentes en memoria.
Población de procesos estática en gran medida.
Poco movimiento de programas entre almacenamiento secundario y memoria.
La gestión de archivos se orienta más a velocidad de acceso que a utilización eficiente del recurso.

Sistemas operativos de red.

La principal función de un sistema operativo de red es ofrecer un mecanismo para transferir archivos de una máquina a otra. En este entorno, cada instalación mantiene su propio sistema de archivos local y si un usuario de la instalación A quiere acceder a un archivo en la instalación B, hay que copiar explícitamente el archivo de una instalación a otra. Internet proporciona un mecanismo para estas transferencias, a través del programa protocolo de transferencias de archivos FTP (File Transfer Protocol).

Suponga que un usuario quiere copiar un archivo A1, que reside en la instalación B, a un archivo A2 en la instalación local A. Primero, el usuario debe invocar el programa FTP, el cual solicita al usuario la información siguiente:

a) El nombre de la instalación a partir de la cual se efectuará la transferencia del archivo (es decir la instalación B).
b) La información de acceso, que verifica que el usuario tiene los privilegios de acceso apropiados en la instalación B.
Una vez efectuada esta comprobación, el usuario puede copiar el archivo A1 de B a A2 en A, ejecutando "get A1 to A2"
En este esquema, la ubicación del archivo no es transparente para el usuario; tiene que saber exactamente donde esta cada archivo. Además los archivos no se comparten realmente, porque

un usuario solo puede copiar un archivo de una instalación a otra. Por lo tanto pueden existir varias copias del mismo archivo, lo que representa un desperdicio de espacio. Así mismo, si se modifican, estas copias no serán consistentes.

Los Sistemas Operativos de red son aquellos sistemas que mantienen a dos o más computadoras unidas a través de algún medio de comunicación (físico o no), con el objetivo primordial de poder compartir los diferentes recursos y la información del sistema.
El primer Sistema Operativo de red estaba enfocado a equipos con un procesador Motorola 68000, pasando posteriormente a procesadores Intel como Novell Netware.
Los Sistemas Operativos de red más ampliamente usados son: Novell Netware, Personal Netware, LAN Manager, Windows NT Server, UNIX, LANtastic.

Operación de los sistemas de cómputo

Un sistema moderno de cómputo de propósito general está compuesto de una cpu y varios manejadores de dispositivos conectados mediante un bus común, que proporciona varios manejadores de dispositivos conectados mediante un bus común, que proporciona acceso a la memoria compartida. Cada controlador está a cargo de un tipo específico de dispositivo. La Cpu y los manejadores de dispositivo pueden operar de manera concurrente compitiendo por ciclos de la memoria. Para asegurar un acceso ordenado a la memoria compartida, se cuenta con un controlador de memoria cuya función es de sincronizar el acceso a la misma.

Para que la computadora empiece a funcionar necesita tener un programa inicial que ejecutar. Este programa inicial o programa de arranque inicial (bootstrap) inicializa todos los aspectos del sistema, desde los registros de la cpu y los controladores de dispositivos, hasta los contenidos de la memoria, así como también debe saber como cargar el sistema operativo y comenzar a ejecutar dicho sistema, localizando y cargando en la memoria el kernel del sistema operativo, hasta que ocurra algún evento.

La ocurrencia de un evento generalmente está señalada por una interrupción, ya sea de hardware o de software. El software puede activar una interrupción ejecutando mediante una operación llamada al sistema, existiendo muchos tipos diferentes de eventos que pueden activar una interrupción , citando un ejemplo:

Terminación de E/S
Una división entre cero
Algún suceso inválido a la memoria ó la solicitud de algún servicio del sistema operativo.

Cuando se interrumpe a la cpu, ésta detiene lo que está haciendo y de inmediato transfiere la ejecución a una localidad fija. La localidad fija comúnmente contiene la dirección inicial de la rutina de servicio para la interrupción

Las interrupciones son una parte importante den la arquitectura de una computadora. El diseño de cada computadora incluye su propio mecanismo de interrupción, pero existen varias funciones comunes. La interrupción debe transferir el control a la rutina de servicio apropiada.

Los sistemas operativos modernos son activados por interrupciones. Si no existen procesos por ejecutar, ni dispositivos E/S que requiera servicio, ni usuarios a quienes responder, un sistema operativo en guardia, esperando a que algo ocurra.

Una trampa (o una excepción) es una interrupción generada por el software, debida a un error

Estructura de E/S

Un controlador de dispositivo mantiene un almacenamiento de buffer local y un conjunto de registros de propósito especial. El controlador es responsable de mover los datos entre los

dispositivos periféricos que controla y su buffer local; el tamaño del buffer local varía de un controlador a otro.

Interrupciones de E/S

Para uncirá una operación de E/S, la CPU carga los registros apropiados dentro del manejador de dispositivos. A su Ves, el controlador examina los contenidos de estos registros par determinar qué acción tomar., si encuentra una solicitud de lectura, el controlador iniciará la transferencia de datos desde el dispositivo hasta su buffer local, una vez que se completa la transferencia de datos le informa al cpu que ha terminado su operación. Esta comunicación la logra activando una interrupción.

Una vez que se inicia la operación E/S ha dos caminos a seguir: se inicia la operación E/S luego, al terminarla, el control se regresa al proceso del usuario. Este caso es conocido como E/S síncrona. La otra posibilidad E/S asíncrona, regresa el control al programa del usuario sin esperar que se complete la operación E/S, esta operación puede continuar mientras ocurren otras operaciones al sistema.

También es necesario que se pueda llevar un registro de varias solicitudes de E/S al mismo tiempo. Para este propósito de E/S: la tabla de estado de los dispositivos indicando el tipo de dispositivo, su dirección y estado. En caso que este ocupado, el sistema operativo también mantendrá una cola de esperas. Por cada dispositivo E/S.

Estructura DMA

Por sus siglas como acceso directo a memoria par dispositivos E/S de alta velocidad. Después de preparar los buffers, apuntadores y contadores para el dispositivo de E/S, el controlador del dispositivo transfiere un bloque completo de datos directamente desde su propio buffer a la memoria, o viceversa, sin la intervención de la cpu, de esta manera, solo se genera una interrupción por bloque, en vez de una interrupción por byte

Estructura de almacenamiento.

Los programa deben estar en la memoria principal denominada memoria de acceso aleatorio, para ser ejecutados, la memoria principal es la única área de almacenamiento a la que el procesador puede tener acceso directamente. La interacción se logra mediante una secuencia de instrucciones loadsore a direcciones específicas de memoria. La instrucción load mueve una palabra desde la memoria principal a un registro interno dentro del cpu, en tanto que store mueve el contenido de un registro a la memoria principal.

Primero traerá una instrucción de la memoria y la almacenará en el registro de instrucción. La instrucción es decodificada y puede hacer que se traigan operando de la memoria y se almacenen en un registro interno. Después que se ha ejecutado la instrucción sobre los operandos, el resultado puede ser almacenado nuevamente en la memoria.

Jerarquía de almacenamiento

1. REGISTROS
2. CACHE
3. MEMORIA PRINCIPAL
4. DISCO ELECTRONICO
5. DISCO MAGNETICO
6. DISCO OPTICO
7. CINTAS MAGNETICAS

Además de tener diferentes velocidades y costros, los sistemas de almacenamiento puede ser volátiles o no. El almacenamiento volátil pierde su contenido cuando se interrumpe la energía del dispositivo.

Almacenamiento en cache

El almacenamiento en cache es un principio importante en los sistemas de cómputo. La información se mantiene normalmente en algún sistema de almacenamiento como la memoria principal. A medida que se utiliza se copia en un sistema de almacenamiento mas rápido. La caché. Bajo una base temporal. Cuando se necesita una pieza particular de información primero verifica si esta e la caché. Si se encuentra, se usa la información de ahí mismo si no es así, se utiliza la localizada en el sistema de almacenamiento principal, colocando una copia en la caché bajo la suposición de que existe una alta probabilidad de que volverá a necesitarse

II. PROCESOS

Un proceso es cualquier actividad que realiza un procesador. Dado que un ordenador puede llevar a cabo distintos procesos simultáneamente, el procesador debe contemplar la posibilidad de ser compartido, lo que se consigue aplicando un algoritmo de planificación.

Informalmente un proceso es un programa en ejecución. Un proceso es mías que el código del programa, el cual también es conocido como la sección del texto., también incluye la actividad actual, representada por el valor del contador de programa y el contenido de los registros del procesador. Un proceso por lo general también incluye la pila del proceso, que contiene datos temporales (parámetros de método, direcciones de retorno y variables locales) y una sección de datos. Que contiene variables globales.

Un programa por sí mismo no es un proceso; un programa es una entidad pasiva, tal como el contenido de un archivo almacenado en disco, en tanto que un proceso es una entidad activa, con un contador de programa que especifica la siguiente instrucción a ejecutarse y un conjunto de recursos asociados.

Aunque dos procesos pueden estar asociados al mismo programa, se les considera como dos secuencias de ejecución distintas.

Se pueden considerar cinco características definitorias de los procesos

- Concurrencia:

Que es la progresión de dos o más procesos que trabajan en paralelo, pero de forma dependiente. Esta es la situación más típica en los sistemas actuales. Por ejemplo, mientras se realizan las operaciones necesarias para la selección e inserción de los campos para preparar las etiquetas que servirán para enviar las facturas de una sucursal bancaria, pueden irse leyendo los datos relativos a vencimiento de talones o preparar listados de riesgo por clientes (. Si la relación es de paralelismo pero independiente, se denomina una relación de convivencia

- Competencia:

Situación que se plantea cuando dos o más procesos deben utilizar los mismos recursos físicos o lógicos. Esto ocurre por que no es posible que dos procesos actúen usando el mismo recurso

- Cooperación:

Lo que ocurre cuando dos procesos que se están ejecutando son interdependientes, es decir, para que lleguen a un resultado, se necesitan mutuamente

- Jerarquía:

Es la relación que se establece entre procesos que son interdependientes en mayor o menor grado, estableciéndose familias de procesos. Cuando un proceso necesita crear otros, al previo se le denomina proceso padre y al nuevo se le llama proceso hijo. Las normas de estado de los procesos en su creación o suspensión las dicta el sistema operativo

- Estado:

Es el grado de uso de la CPU. Existen tres niveles:

Nuevo: el proceso se esta creando
En ejecución : se están ejecutando instrucciones
En espera: el proceso está esperando que ocurra algún evento.
Listo: el proceso está en espera de ser asignado a un procesador.
Terminado: el proceso ha terminado su ejecución

Características:

- Un proceso consta de código, datos y pila.
- Los procesos existen en una jerarquía de árbol (varios Hijos, un sólo padre).
- El sistema asigna un identificador de proceso (PID) único al iniciar el proceso.
- El planificador de tareas asigna un tiempo compartido para el proceso según su prioridad (sólo root puede cambiar prioridades).
- Área de usuario: está en el kernel y consiste de la tabla de descriptores de archivos, información sobre el consumo de CPU, manejadores de señales, entre otros.
- Tabla de páginas

Tipos de procesos

Son diferentes según las características y/o atributos que poseen. Siguiendo el criterio de la utilización de la memoria, los procesos pueden ser:

- Proceso residente: Se trata de aquel proceso que durante su estado de activado tiene que estar cargado en la memoria
- Proceso intercambiable: Como aquel proceso que está en estado de espera y del cual se toma su proceso tratado por la CPU y se incluye en un fichero del sistema, con lo que se deja liberada la porción de memoria correspondiente

Si se sigue el criterio de atender a las características de su código, los procesos serán:

- Procesos reutilizables: O programas que pueden usar varios usuarios ya que están codificados con variables genéricas y los usuarios sólo tienen que introducir los valores correspondientes a la posición de las variables en el código. Tiene la ventaja de que se carga en memoria una sola vez y se usa varias veces inicializando adecuadamente las variables

- Procesos reentrantes: Procesos cuyas instrucciones son invariantes, con lo que pueden usarse de nuevo sin tener que cargarse otra vez. Están estructurados separando lógicamente los segmentos de datos y de código, con lo que dos instancias del mismo proceso pueden compartir el mismo código
- Proceso de excepciones: Circunstancias que se pueden presentar en un proceso ocasionadas por un suceso que se ha presentado pero que no debería haber tenido lugar, con lo que el sistema tratará de eludirlo. Es lo que se denomina comúnmente un error

Colas de Planificación

A medida que los procesos van entrando al sistema, se les coloca en una cola de trabajos en están todos los procesos en el sistema. Los procesos que residen en la memoria principal listos y en espera de ejecutarse se mantienen en una lista denominada la cola de procesos listos.

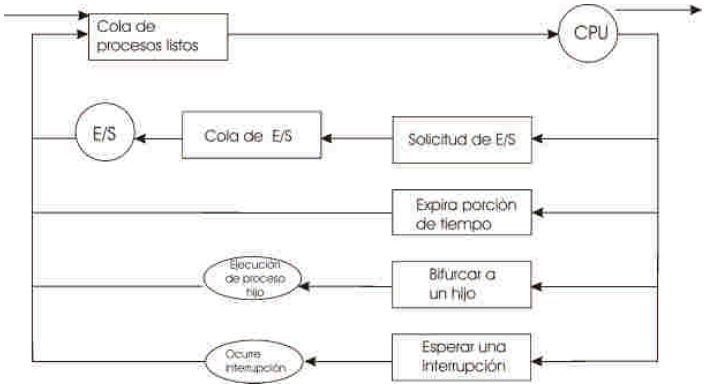
Esta cola se almacena por lo general como una lista enlazada. El encabezado de cola de procesos listos contiene apuntadores al primero y al último PCB en la lista. Cada PCB se extiende para incluir un campo apuntador que señala al siguiente PCB en la cola de procesos listos.

Existen otras colas en el sistema. Cuando a un proceso se le asigna la CPU, se ejecuta durante cierto tiempo; después abandona, es interrumpido, o debe esperar la ocurrencia de algún evento particular, como sería la terminación de una solicitud E/S. En el caso de una solicitud de E/S, tal petición puede ser una unidad de cinta dedicada, o a un dispositivo compartido, como un disco. Debido a que hay muchos procesos en el sistema, el disco puede estar ocupado con la solicitud de e/s de otro proceso. Por lo tanto, el proceso tiene que esperar por el disco. La lista de procesos que están esperando por un dispositivo de E/S particular se conoce como cola del dispositivo. Cada dispositivo tiene su propia cola.

La representación común para estudiar la planificación de procesos es un diagrama de colas. Cada rectángulo representa una cola; se muestran los tipos de colas: la cola de procesos listos y un conjunto de colas de dispositivos. Los círculos representan los recursos que dan servicio a las colas y las flechas indican el flujo de procesos en el sistema.

Un nuevo proceso nuevo inicialmente se coloca en la cola de procesos listos, donde espera hasta que es seleccionado para su ejecución o es despachado. Una ves que se asigna la cpu al proceso y se está ejecutando, puede ocurrir uno de varios elementos:

El proceso emite una solicitud de E/S, y entonces es colocado en la cola de E/S.
El proceso crea un nuevo subproceso y espera su terminación.
El proceso es removido forzosamente de la CPU, como resultado de una interrupción, y es colocado de nuevo en la cola de procesos listos.



Administración de la memoria principal

El sistema operativo es responsable de las siguientes actividades relacionadas con la administración de la memoria

- Llevar un registro de las partes de la memoria que están siendo utilizadas en el momento y quién las está utilizando.
- Decidir qué procesos se van a cargar en la memoria cuando el espacio de la misma esté disponible.
- Asignar y liberar espacio de la memoria según se necesite

Administración de archivos

La administración de archivos es uno de los componentes más visibles en un sistema operativo. Las computadoras pueden almacenar información en varios tipos diferentes de medios físicos.

Para un uso conveniente del sistema de cómputo el sistema operativo proporciona una vista lógica uniforme del almacenamiento de la información. Este sistema hace una abstracción de la propiedades físicas de sus dispositivos de almacenamiento para definir una unidad de almacenamiento lógico , el archivo. El sistema operativo hace un mapa de los archivos en los medios físicos y accede a estos archivos vía los dispositivos de almacenamiento.

Un archivo es un conjunto de información relacionada definida por su creador, los archivos representan programas tanto en forma de fuente como de objeto y datos.

El sistema operativo es responsable de las siguientes actividades relacionadas con la administración de archivos.

Crear y eliminar archivos
Crear y eliminar directorios
Dar soporte a primitivas para la manipulación de archivos y directorios.
Hacer un mapa de los archivos en el almacenamiento secundario.
Respaldar archivos en medios de almacenamiento estables (no volátiles).

Administración del sistema E/S

Uno de los propósitos de un sistema operativo es ocultar al usuario las peculiaridades de los dispositivos específicos de Hardware, el subsistema de E/S oculta las particularidades de los dispositivos E/S al resto del sistema operativo mismo.

El subsistema de E/S consta de:

- Un componente de administración de memoria que incluye manejo de buffers, asignación de cache.
- Una interfaz general de manejadores de dispositivo.
- Controladores para dispositivos de Hardware específicos.

Administración de almacenamiento secundario.

El almacenamiento secundario es como respaldo de la memoria principal. La mayoría de los sistema de cómputo modernos emplean discos como el principal medio de almacenamiento en línea, tanto para programas como para datos.

El sistema operativo es responsable de las siguientes actividades relacionadas a la administración de discos:

- Administración de espacio libre.
- Asignación de almacenamiento
- Planificación del disco

Operación en red

Un sistema distribuido es un conjunto de procesadores que no comparten memoria, dispositivos periféricos o un reloj. En su lugar, cada procesador tiene su propia memoria y reloj local y los procesadores se relacionan entre ellos mediante varias líneas de comunicación como buses de alta velocidad o redes. Los procesadores en un sistema distribuido varían en tamaño y en función: pueden incluir pequeños microprocesadores, estaciones de trabajo, minicomputadoras y sistemas de cómputos grandes de propósitos generales.

Un sistema distribuido reúne sistemas físicos distintos, posiblemente heterogéneos, en un solo sistema coherente, proporcionando al usuario el acceso a los diversos recursos que el sistema mantiene. El acceso a un recurso compartido permite una aceleración del cómputo, mayor funcionalidad y disponibilidad de los datos y una mejora en la confiabilidad.

Sistema de intérprete de comandos.

Uno de los programas del sistema más importante para un sistema operativo es que el intérprete de comandos, que es la interfaz entre el usuario y el sistema operativo, muchos comando se transmiten al sistema operativo mediante sentencias de control. Cuando se inicia un trabajo nuevo en un sistema por lotes, o cuando un usuario se conecta a un sistema de tiempo compartido, se ejecutan automáticamente un programa que lee e interpreta estas sentencias de control. Este

programa también se conoce como intérprete de tarjetas de control o intérprete de línea de comandos, y con frecuencia se denomina shell, Su función es simple: obtener la siguiente sentencia de comandos y ejecutarla.

Servicio del sistema operativo.

- Ejecución de programas
- Operaciones de E/S
- Manipulación del sistema de archivo.
- Comunicaciones
- Detección de errores
- Asignación de recursos
- Contabilidad
- Protección

Llamadas al sistema

Las llamadas al sistema proporcionan la interfaz entre un proceso y el sistema operativo, estas llamadas están disponibles como instrucciones en lenguaje ensamblador y comúnmente se incluyen en los manuales empleados por los programadores de estos lenguajes.

Un enfoque consiste en que el programa pida al usuario los nombre de los dos archivo, este enfoque requerirá una secuencia de llamadas al sistema, primero para escribir un mensaje en la pantalla que pida información y luego para leer del teclado los caracteres que definen a los dos nombres, una ves que se obtienen los dos nombres del archivo, el programa debe abrir el archivo de entrada y crear el de salida. Cada una de estas operaciones requiere de otras llamadas al sistema.

Las llamadas al sistema se pueden agrupar de manera general en cinco categorías.

- control de procesos
- manipulación de archivos
- manipulación de dispositivos
- mantenimiento de información y comunicaciones

Administración de dispositivos.

Un programa , al esta en ejecución, tal vez necesite recursos adicionales para continuar, estos pueden ser más memoria, unidades de cinta, acceso a archivos etc. Si los recursos están disponibles se pueden conceder, y devolver el control al programa del usuario; en caso contrario, el programa tendrá que esperar hasta que se cuente con los recursos suficientes

Comunicación

Existen dos modelos de comunicación.

En el modelo de paso de mensajes la información se intercambia mediante un servicio de comunicación entre procesos proporcionando por el sistema operativo

En el modelo de memoria compartida los procesos utilizan las llamadas al sistema para obtener acceso a regiones de memoria que corresponden a otros procesos, la memoria compartida requiere de dos o más procesos que estén de acuerdo en remover esta restricción y entonces poder intercambiar información leyendo y escribiendo datos en estas áreas compartidas.

Enfoque por capas

El enfoque por capas en el sistema operativo se descompone en varios estratos o niveles cada uno de ellos construido sobre capas inferiores. La capa inferior es el hardware es la mas alta interfaz del usuario. Una capa del sistema operativo es una implementación de un objeto abstracto, que es encapsulamiento de datos y de las operaciones que pueden manipular dichos datos.

Las capas se seleccionan de tal manera que cada una utilice funciones y servicios exclusivamente de capas de niveles inferiores. Este enfoque simplifica la depuración y la verificación del sistema, cada capa se implementa sólo con aquellas operaciones proporcionadas por capas de nivel inferior. Una capa no necesita sabe cómo están implementadas estas operaciones; sólo requiere conocer qué hacen tales operaciones. Por lo tanto, cada capa oculta a las capas de nivel superior la existencia de ciertas estructuras de datos, operaciones y hardware.

Microkernels

La función principal del microkernel es proporcionar un módulo de comunicaciones entre el programa cliente y los diversos servicios que también están ejecutándose en el espacio del usuario.

Máquinas virtuales

Es crear la ilusión de que un proceso tiene su propio procesador con su propia memoria virtual.

Máquina virtual

La máquina virtual Java es una especificación para una computadora abstracta. La JVM consta de un cargador de clases y un intérprete de Java que ejecuta los códigos de bytes independientes de la arquitectura.

III. SINCRONIZACIÓN Y COMUNICACIÓN ENTRE PROCESOS

Bloque de control del proceso

Cada proceso se representa en el sistema operativo mediante un bloque de control del proceso, también denominado bloque de control de tarea, este contiene diversas piezas de información asociadas a un proceso específico incluyendo:

Estado del proceso: el estado puede ser nuevo, listo en ejecución, espera, detenido etc..

Contador de programa: el contador indica la dirección de la siguiente instrucción a ejecutar.

Registros del CPU: incluyen acumuladores, registros, índice, apuntadores de pila y registros de propósito general, además de cualquier información de código de condición. Junto con el contador de programa, esta información de estado debe guardarse cuando ocurre una interrupción, para posteriormente permitir que el proceso continúe de forma más apropiada.

Información de planificación del CPU: esta información incluye la prioridad del proceso, apuntadores de colas de planificación y cualquier otro parámetro de planificación.

Información de administración de la memoria: esta información puede incluir datos referentes al valor de los registros base y límite, las tablas de páginas, o las tablas de segmentos, dependiendo del esquema de memoria empleado por el sistema operativo.

Planificadores

Un proceso migra entre las diversas colas de planificación a lo largo de su existencia para fines de planificación, el sistema operativo debe seleccionar en alguna forma los procesos de estas colas. Esta selección la realiza el planificador apropiado.

El planificador de largo plazo o planificador de trabajos, selecciona procesos de esta reserva y los carga en memoria para su ejecución. La distinción principal entre el planificador de largo plazo y el planificador de corto plazo, es la frecuencia de su ejecución, el planificador de corto plazo debe seleccionar repetidamente un nuevo proceso para que la cpu, ejecute durante solo unos cuantos milisegundos antes de esperar una solicitud de esa, por su parte el planificador de largo plazo se ejecuta con menos frecuencia, y puede pasar minutos a la creación de un nuevo proceso.

Un proceso limitado por E/S es el que consume más de su tiempo haciendo operaciones E/S que haciendo cálculos.

Un proceso limitado por CPU es el que genera solicitudes de E/S de manera poco frecuente empleando más tiempo en cálculos que utilizando por un proceso limitado de E/S.

Conmutación de contexto

La conmutación de la cpu a otro proceso requiere conservar el estado del proceso anterior y cargar el estado guardado del nuevo proceso incluyendo el valor de los registros del la cpu y el estado del proceso y la información sobre administración de la memoria, esto solo implica cambiar el apuntador al conjunto de registros actual.

Procesos cooperativos.

Un proceso cooperativo puede verse afectado por los otros procesos que están ejecutando ene l sistema; cualquier proceso que comparte datos con otros procesos es un proceso cooperativo.

Un proceso productor genera infamación que es utilizada por un proceso consumidor, para permitir que los procesos productores y consumidores se ejecuten concurrentemente, se debe de tener un buffer de elementos que pueda ser llenado por el productor y vaciado por el consumidor.

Sincronización

La comunicación entre procesos tiene lugar mediante llamadas a las primitivas send y recive. Existen diferentes opciones de diseño para implementar cada primitiva. El paso de mensajes puede ser con bloque o sin bloqueo también conocidos como síncrono y asíncrono.

- Envió con bloqueo: el proceso emisor se bloquea hasta que el mensaje es recibido por el proceso receptor o por el buzón.
- Envío sin bloqueo: el proceso emisor envía el mensaje y continúa su operación
- Recepción con bloqueo: el receptor se bloquea hasta que haya un mensaje disponible.
- Recepción sin bloqueo: el receptor recupera un mensaje válido, o bien un valor nulo.

Hilos

Un hilo, denominado también proceso ligero, es una unidad básica de utilización de la CPU; comprende la ID del host, un contador de programa, un conjunto de registros y una pila. El hilo comparte con otros hilos que pertenecen al mismo proceso su sección de código, su sección de datos y otros recursos del sistema operativo.

Beneficios:

- Su grado de respuesta:
- Compartir recursos
- Economía
- Utilización de arquitecturas de multiprocesadores.

Hilos de usuario

Los hilos de usuario tienen soporte por encima del kernel y son implementados por una biblioteca de hilos a nivel usuario. La biblioteca proporciona apoyo para la creación, programación y administración de hilos sin soporte del kernel.

Hilos de kernel

Los hilos de kernel tienen soporte directo del sistema operativo, la creación, programación y administración de hilos los realiza el kernel en su propio estado.

Modelos de multihilos

Modelo de varios a uno: Mapea múltiples hilos a nivel de usuario en un hilo de kernel

Modelo de uno a uno: Mapea cada hilo de usuario e un hilo de kernel.

Modelo de varios a varios: Combina muchos hilos a nivel de usuario con un número menor o igual de hilos de kernel

Planificación de la CPU

La planificación de la cpu es la tarea de seleccionar de la cola de listos un proceso que esta en espera y asignarle el procesador.

El despachador asigna la cpu al proceso seleccionado.

La planificación del tipo primero en llegar, primero en ser atendido es el algoritmo de planificación más sencillo, pero puede ocasionar que los procesos cortos tengan que esperar procesos muy largos.

Se puede probar que la planificación de primero el trabajo más corto es óptima, proporcionando el tiempo de espera promedio más corto. La implementación de la planificación SJF es difícil debido a que no es fácil predecir la duración de la siguiente ráfaga de cpu. El algoritmo SJF es un caso especial del algoritmo general de planificación por prioridades, el cual simplemente asigna la cpu al proceso mayor prioridad. Tanto la planificación por prioridades como la planificación SJF pueden sufrir inanición.

La planificación round-robin (RR) es más apropiada para un sistema de tiempo compartido. La planificación RR asigna la cpu al primer proceso en la cola de listos durante q unidades, en donde q es el quantum. Si el quamtum es demasiado largo, la planificación RR degenera a una planificación FCFS; si el quantum es demasiado pequeño, el trabajo adicional de planificación en la forma de tiempo de conmutación de contexto se vuelve excesivo.

El algoritmo FCSFS no es apropiativo; el algoritmo RR es apropiativo. Los algoritmos SJF y por prioridades pueden ser apropiativos o no apropiativos .

Loa algoritmos de colas de niveles múltiples permiten utilizar diferentes algoritmos para varias clases de procesos. El más común es una cola interactiva de proceso de primer plano que utiliza la planificación RR, y una cola en lotes de procesos de segundo plano, que utiliza la planificación FCFS. Las colas de niveles múltiples son retroalimentación permiten que los procesos se muevan de una cola a otra.

Muchos sistema de computo actuales ahora tienen soporte para múltiples procesadores, en donde cada procesador se planifica a sí mismo de manera independiente. Normalmente, hay una cola de procesos ó hilos, todos los cuales están disponibles para ejecución. Cada procesador una decisión de planificación y selecciona un proceso de esta cola.

La JVM utiliza un algoritmo de planificación de hilos apropiativo basado en prioridades, que selecciona para su ejecución al hilo con prioridad máxima. La especificación no indica si la JVM deberá asignar porciones de tiempo a los hilos; eso depende de la implementación en particular de la JVM.

Sección critica

Cada hilo tiene un segmento de código, denominado sección crítica, en el que el hilo puede estar modificando variables comunes, actualizando una tabla, escribiendo un archivo..La característica importante del sistema es que, cuando un hilo se está ejecutando en su sección crítica, no se debe permitir que otros hilos se ejecuten en esa misma sección. Por lo tanto la ejecución de secciones criticas por los hilos es mutuamente exclusiva en el tiempo.

Exclusión mutua.

Si el hilo T1 se está ejecutando en su sección crítica, entonces ningún otro hilo puede estar en la misma ejecución.

Sincronización

Existen relaciones entre los procesos de forma que hasta que uno de ellos no ha acabado, otro no puede comenzar. Para que un conjunto de procesos funcione adecuadamente muchas veces será necesario sincronizar sus actividades para garantizar que ciertos procesos terminan antes de que comiencen otros.

Para poder hacer esto, el sistema recurre al envío de señales. Un proceso permanece en espera de recibir ciertas señales, por lo que está suspendido. Cuando las señales llegan, se pone en estado de ejecución. Es corriente que entre los procesos implicados se produzca una relación de cooperación para la ejecución de tareas comunes y una relación de competencia por el uso de recursos compartidos.

Por ejemplo, si se ejecutan los siguientes procesos:

Proceso P_i	Proceso P_j
$Y_1 = X$ $Y_1 = X + 1$ $X = Y$	$Y_2 = X$ $Y_2 = Y_2 + 1$ $X = Y_2$
Ejecución (X = 0)	
$Y_1 = X$ $Y_1 = X + 1$ $X = Y_1$	$Y_2 = X$ $Y_2 = X + 1$ $X = Y_2$
Conclusión: X = 1, valor incorrecto para P_i	

Con lo que el sistema operativo utilizará una variable especial intermedia para que la información común a ambos procesos sea bloqueada en P_i mientras la utiliza el primer proceso -si es que éste es el que tiene la prioridad- mientras el proceso P_j espera. Más tarde éste recibe el valor de X y se ejecuta proporcionando el verdadero valor de X para ese proceso, que será igual a 2.

Exclusión mutua

Para evitar la concurrencia de los procesos y optimizar el uso de la CPU aprovechando los tiempos muertos o momentos valle, se ideó el método de la exclusión mutua, para el caso de sistemas multiprogramados.

Cuando dos procesos se encuentran compartiendo una variable, se crean conflictos dado que el valor de la variable cambiará según el proceso que la utilice. Por ello se hace necesario asegurarse de que los dos procesos nunca van a ejecutar su sección crítica -es decir, aquella parte de sus instrucciones que utilizan esa variable compartida- al mismo tiempo. Así se evitan riesgos de error, colisiones, esperas y no estar sujeto a la dependencia de la velocidad de los procesos ni del número de procesadores.

La solución denominada exclusión mutua consiste en permitir el uso de la variable compartida al proceso que tiene prioridad mientras los otros procesos permanecen en espera. Para ello habrá que indicar una serie de condiciones para que los procesos en espera no se vean interrumpidos o suspendidos. Por ese motivo se crean diferentes algoritmos de exclusión, utilizando mecanismos de hardware y soluciones de software, y dentro de éstas aparecen las soluciones de espera activa y de espera no activa.

Métodos de software

Algoritmos elementales de exclusión mutua

Se trata de inventar una variable que sirva para todos los procesos relacionados y que recoge un valor en función de la posibilidad de que un proceso entre en ejecución de su sección crítica. Cuando el proceso se ha ejecutado, devuelve a esa variable un valor que permitirá a la CPU dejar paso libre para que otro proceso entre en ejecución.

Esto tiene la desventaja de que se realiza el paso de unos procesos a otros sin tener en cuenta ni las prioridades ni la conveniencia de trabajo eficiente del sistema. Además, si el primer proceso

sufriera una interrupción, el resto de los procesos no podrían disponer de las variables compartidas o no podrían actuar conforma a la variable que gobierna todo el proceso.

Para evitar este problema se inventaron una serie de variables, tantas como procesos concurrentes, que se adjudican a los procesos, una a uno, que servirá para que cada proceso señale, en esa variable, a la CPU que se encuentra ejecutando su sección crítica. Así, cada proceso consulta las variables de los demás procesos para "estimar" la conveniencia de entrar en ejecución de la sección crítica o no.

Sin embargo, las prioridades de este método no están claras y pueden producirse cuellos de botella o infrautilizaciones de la CPU.

- Algoritmo de Dekker

Se trata de la primera solución conocida de tipo software para resolver la concurrencia de procesos. Matemáticamente está basado en interpolaciones lineales sucesivas y búsquedas binarias capaces de encontrar el cero de una función que cambia de signo en un intervalo dado.

Es un algoritmo de espera activa, es decir, cuando las instrucciones de un proceso van preguntando cada vez a la CPU si el proceso que se está ejecutando en paralelo y que tiene una región crítica a terminado ya o tiene que seguir esperando. Como ejemplo de un algoritmo de Dekker, sería el siguiente, considerando los subíndices i y j como diferentes procesos:

```
<RUTINA>
INICIALIZA ACTIVO (i) = V
        MIENTRAS ACTIVO (j) = V <NADA>
                SI TURNO = j ↦ ACTIVO (i) = F
                        MIENTRAS TURNO = j <NADA>
                        ACTIVO (i) = V
END
TURNO = j
ACTIVO (i) = F
```

Si bien existe la posibilidad de que se entre en un bucle sin fin, por que podría pasar que:

```
ACTIVO (i) = V
ACTIVO (j) = V
```

- Algoritmo de Petterson

```
<RUTINA>
INICIALIZA ACTIVO (i) = V
                                TURNO = j
        MIENTRAS ACTIVO (j) = V y TURNO = j <NADA>
END
ACTIVO (i) = F
```

Para el proceso j el algoritmo de exclusión sería idéntico.

- Algoritmo de Dijkstra o del semáforo

Se trata de un método diseñado por E. W. Dijkstra en 1959 para tratamiento gráfico y que tenía como objetivo encontrar el camino más corto entre un vértice y los restantes, dentro de los llamados gráficos ponderados, o gráficos que llevan asociada a cada arista -línea que une dos

vértices- una cantidad representativa de un concepto. Por ejemplo, un gráfico de rutas de viaje, donde cada arista une dos puntos geográficos y lleva asociado un costo de viaje.

Consta de dos operaciones que trabajan coordinadamente: señal y espera. Estas señales actúan sobre una variables que por la función que realiza se llama semáforo. Esta variable, que es compartida, toma valores enteros no negativos. Si se da el caso de que toma sólo los valores 0 y 1 entonces a la variable se la llama semáforo binario. Los valores que toma la variable es el número de procesos que pueden entrar a la vez en su sección crítica. El pseudocódigo para este algoritmo sería el siguiente:

```
<SEMÁFORO-ENTRAR>
INICIALIZA SEMAFORO = VALOR
SI SEMAFORO > 0 ↦ SEMAFORO = SEMAFORO - 1
        SECCIÓN CRÍTICA
        SI NO <ESPERA>
<SEMÁFORO-SALIR>
SI SEMAFORO <= 0 ↦ SEMAFORO = SEMAFORO + 1
        DESBLOQUEAR SIGUIENTE PROCESO
        SI NO SEMAFORO = SEMAFORO + 1
FIN
```

(si SEMAFORO fuese igual a 0, entonces no admitiría más procesos en sección crítica)

Si se añade una entidad denominada cola de espera, se evitará que algún proceso entre en una dinámica de espera indefinida.

- Algoritmo de Camport o de la tienda

Se trata, como ocurre en algunas tiendas, de proporcionar a cada proceso un número de “cliente”, dando entrada en la sección crítica a aquél que tenga el número más bajo, y así sucesivamente. Sin embargo, además de presentarse problemas de infrautilización de la CPU, puede darse el caso de que dos procesos tengan acceso al mismo número, con lo que se crearía el mismo conflicto que se quiere evitar.

Mecanismos de hardware

Para evitar el caso de que los métodos de software fallen, lo que podría ocurrir cuando se presenta un fallo de programación -lo cual no es infrecuente si se tiene en cuenta que el lenguaje de algunos de estos algoritmos será el ensamblador-, se diseñaron mecanismos de hardware, normalmente codificados en lenguaje de bajo nivel y decodificados directamente por la CPU.

Interrupción permitida/interrupción no permitida

Viene a actuar respecto a los procedimientos de la CPU como si se tratase de un interruptor. Cuando la CPU recibe desde un proceso la instrucción interrupción deshabilitada -DI, del inglés disable interrupt, la CPU queda bloqueada e ignora las interrupciones y ejecuta en exclusiva las instrucciones del proceso que se la mandó. Es decir, este mandato detiene todos los procesos, aunque no tengan secciones críticas con respecto del proceso emisor. Cuando el proceso ha terminado de ejecutarse debe volver a habilitarla porque si no el procesador permanecería en un bucle dado que estaría continuamente ejecutando el proceso emisor.

En un sistema multiproceso, la llegada de la instrucción de bloqueo a una parte del sistema no influye en las otras partes del sistema que están procesando, con lo que no se tiene una garantía de coherencia en los resultados obtenidos.

Comparar y asignar

Su mecanismo de trabajo es similar al método software del semáforo, pero implantado a nivel hardware. Utiliza un bit de acceso para cada zona de datos definida -zonas que sufren de concurrencia de procesos y que su actualización afecta a los resultados de cada proceso-, de tal manera que cuando el valor de ese bit es 1, la zona se bloquea y sólo puede utilizarla el proceso actual. Si el valor es 0, permite la entrada y actualiza el bit a 1 para que ningún otro proceso pueda acceder a esa zona.

Es bastante común el uso de este mecanismo en sistemas multitarea y multiusuario. Se suele usar cuando pueden presentarse muchas situaciones de conflicto.

Variable de control

Trata de controlar cuando un proceso quiere actualizar una variable en conflicto -caso de una variable compartida- y averiguar si otro proceso anterior lo había hecho previamente. En el caso de que haya sido así, el proceso actual lo intentará hasta que el sistema le permita a ello y proseguir con su ejecución. Para ello, este mecanismo compara el valor que tenía la variable conflictiva al empezar la sección crítica y si no es el adecuado, lo intercambia por su inicialización.

Es un buen método cuando se trata de sistemas en los que no se prevé gran número de conflictos.

Algoritmos generales de planificación de la CPU

- Método FIFO o FCFS

FCFS es lo que se conoce como “primero en llegar, primero en ser servido” –del inglés First Come First Served-, equivalente al método FIFO, que se llama así por ser las siglas del término inglés *First In First Out*, es decir, el primero en entrar será el primero en salir. Un sinónimo para este método es el de la *queue list*, o lista donde los añadidos se colocan al final de la lista, y las remociones se producen al principio.

Es decir, se atenderá por la CPU al primer proceso que llega. El resto de los procesos pasan a la cola de espera donde permanecen en un estricto orden de llegada. Suele utilizarse en sistemas donde el costo del servicio es más alto que el costo producido por las congestiones.

El inconveniente más serio es que es impredecible, puesto que no se sabe qué procesos están llegando a la CPU. Además los usuarios esperan que los tiempos de respuesta sean muy similares para procesos parecidos, lo cual no se garantiza con es método, afectando en mayor medida a los procesos breves, sobre todo si van detrás de algún procesos largo. Por otro lado, este método es no apropiativo, es decir, el proceso permanece usando la CPU mientras no ha terminado.

Un ejemplo de este método ha quedado descrito en las tablas 7.1 y 7.2.

- Método Round-Robin

Por este método se define un intervalo de tiempo idéntico –denominado *quantum*- para todos los procesos de la cola, la cual se trata de manera circular.

Es decir, a cada proceso se le asigna un tiempo de uso de la CPU. Si ese tiempo se acaba, aunque el proceso no haya terminado, éste es dirigido a la cola -al final- y otro proceso pasa a ejecutarse. Parece obvio que si los tiempos asignados son muy cortos, el proceso se hará más lento dado que las rutinas de activación y desactivación, así como las instalaciones en cola, se harán muy frecuentes.

Una variante más sofisticada es la cola de retroalimentación -del inglés *feedback queue*- que se usa sobre todo en sistemas multiusuario. Por esta variante, un proceso se ejecuta hasta que acaba su intervalo de tiempo asignado, o hasta que comienza a transferir datos a un periférico o hasta que ocurre alguna interrupción generada por otro proceso. Si el proceso queda inacabado, se le asigna más tiempo que la vez anterior y se le dirige al final de la cola. Lo mismo ocurre si había iniciado una transferencia con un periférico llevándole al principio de la cola.

- Por el proceso más corto

Se trata del método SJF -del inglés *Short Jobs First*, es decir, primero los procesos más cortos-. Por este método se eligen de la cola los procesos más cortos y se les asigna máxima prioridad para el uso de la CPU.

Tiene la ventaja de que aprovecha más eficientemente el tiempo de uso de la CPU, y prioriza los procesos más cortos, lo que conduce a que en una unidad o intervalo de tiempo, se acaben mayor número de procesos.

Las desventajas vienen de que no se sabe bien cómo asigna las prioridades y cómo asigna los tiempos de uso de la CPU a cada proceso, dado que la CPU no quedará libre hasta acabar el proceso en cuestión. Por otro lado, la reasignación de prioridades por tiempos puede ser dinámica, en función del uso detectado de la CPU.

Como el método FIFO, este es no apropiativo, es decir, el proceso permanece usando la CPU hasta que termina.

- Por el proceso más corto y el resto

Este método compara el tiempo de ejecución que le queda al proceso que está utilizando la CPU con el tiempo que necesita el proceso que se encuentra el primero en la cola de espera, que será el más corto de los que están en dicha cola. Si el tiempo del que está en la cola es menor, quita de la CPU el proceso en curso y deja libre la CPU para el que viene de la cola. En resumen, se trata de un método apropiativo, pues quita CPU para dársela a otro proceso.

La eficiencia del uso de la CPU es mayor, pero el sistema operativo tiene que guardar los estados y tiempos de los procesos que salen momentáneamente de la CPU, lo que origina mayor capacidad de almacenamiento y mayor número de desplazamientos, que también consumen recursos y tiempo.

- Prioridad multicola

El sistema operativo analiza los procesos, sus tiempos de ejecución y sus prioridades, construyendo varias colas de espera. Cada cola puede llevar un método diferente de gestión según la prioridad de la misma, o bien ser atendida por un método FIFO o circularmente.

Los sistemas operativos que operan según estas características se denominan sistemas operativos de propósito general y son capaces de soportar varios trabajos interactivos con el usuario, normalmente tantos como procesos batch o procesos compilados para trabajar secuencialmente según su opción de ejecución.

- Por tiempo de espera y de ejecución

Aquí la prioridad la establecen dos parámetros: el tiempo de espera en la cola y el tiempo estimado de ejecución del proceso.

El acceso es no apropiativa y la implementación del código en el sistema resulta costosa y conlleva cierta sobrecarga de memoria.

- Mixtos

El más común es el que supone la mezcla de un criterio Round-Robin con un criterio multicola. Según esto, un sistema asigna tiempos de la CPU mediante diversas colas que se establecen según su nivel de prioridad. Si un proceso pasa de una cola a la CPU y no acaba en el tiempo que se le ha asignado, se le dirige para el resto de su ejecución a la cola siguiente. Si, tras su siguiente ejecución, no ha acabado, entonces se le pasa a la sucesiva. La última cola está gestionada por un método FIFO.

Este sistema sigue dando prioridad a los procesos que son más cortos

Interbloqueos

Es una situación que se plantea cuando dos procesos activos compiten por recursos del sistema y uno de los procesos no consigue los suficientes y entra en estado de espera. Cuando este estado de espera es indefinido se denomina situación de interbloqueo -*deadlock*-.

El proceso P requiere los recursos X e Y y los necesita en ese orden y, simultáneamente, el proceso Q necesita los recursos Y y X y los necesita en ese orden. P adquiere X y, a la vez, Q toma Y. Llegado un momento ninguno de los procesos puede progresar dado que va a requerir recursos que posee el otro proceso. Su solución sólo puede venir del exterior y es una situación típica de sistemas multiprogramados.

Se entiende aquí por recurso tanto dispositivos hardware -caso de cintas, algún periférico- como software -información en módulos de memoria, contenido de un array-.

Para que se de el interbloqueo deben darse simultáneamente cuatro condiciones:

- Exclusión mutua: cuando algún proceso necesita control exclusivo sobre un recurso, forma no compartible, y lo acapara, impidiendo que otro proceso lo pueda utilizar
- Espera y retención: situación que se da cuando un proceso que retiene un recurso lo mantiene hasta que pueda utilizar otro
- No apropiación: un recurso permanece en propiedad de un proceso y no se le puede quitar hasta que lo haya utilizado; es decir, las prioridades tienen una escasa validez
- Espera circular: existe un conjunto de procesos en el que se da la circunstancia de que un proceso retiene uno o más recursos que necesita el siguiente proceso

Soluciones al interbloqueo

La solución más radical es eliminar los procesos que han ocasionado el interbloqueo, lo que se conoce como reanudación. Otras soluciones pueden ser establecer protocolos para evitarlos, asignar prioridades o mantener informado al sistema y al usuario de los estados de procesamiento.

En general existen dos grandes tipos de soluciones, las soluciones preventivas y las de tratamiento o médicas, es decir, las que se aplican una vez que el interbloqueo ha tenido lugar.

Prevención del interbloqueo

Lo más eficaz en este caso es evitar que alguna de las condiciones que da lugar al interbloqueo no ocurra. Para ello las vemos una por una:

- Exclusión mutua: Resulta complicado ya que existen recursos que no se pueden compartir, como es el caso de los dispositivos de entrada y de salida -las cintas y las impresoras, por ejemplo-, con lo que se tiene que recurrir a una compartición virtual con el uso del denominado *spooler*.

- Espera y retención: Se evita si el sistema operativo hace que cada proceso, antes de entrar en ejecución, muestre los recursos que va a utilizar declarándolos en el Lenguaje de Control de Tareas -JCL, de *Job Control Language*-. El JCL es un lenguaje que se utiliza para elaborar "macros" que contienen la secuencia de comandos que se van a ejecutar cuando un programa entra en estado operativo. Si el sistema operativo comprueba que todos los recursos declarados están disponibles, le permitirá la ejecución. Si no, le pondrá en estado de preparado hasta que se liberen esos recursos.
  Una desventaja es que los recursos tienen una baja utilización dado que algunos de los recursos que se usarán al final de la ejecución de un proceso no se utilizarán hasta entonces permaneciendo libres. Además se pueden discriminar procesos postergando algunos en el tiempo más de lo debido.
- No apropiación: El sistema operativo estudia si un proceso solicita un recurso que no está disponible y, en ese caso, lo mantendrá en estado preparado liberando los recursos que previamente tenía asignados. Este método puede postergar indefinidamente algunos procesos y puede desechar trabajo que ya ha sido realizado.
- Espera circular: El programador deberá especificar en cada proceso el orden en el que va a solicitar los recursos, estableciendo así una prioridad de los mismas para cada proceso. De tal manera que si un proceso está utilizando el recurso situado en el orden 3 -por ejemplo, un fichero- y quiere usar un recurso situado en primer lugar -por ejemplo, una cinta-, el proceso liberará los restantes recursos y se colocará en el primer lugar del orden de uso. Para ir pasando de un lugar al inferior deben irse liberando los recursos superiores.

Como se ha visto en los puntos anteriores, pude ocurrir que la aplicación de medidas para prevenir el interbloqueo dé lugar a una baja utilización de los recursos. Para evitar esta circunstancia es conveniente que el sistema y el usuario dispongan de información sobre cómo se van a solicitar más recursos y en qué orden.

Eliminación del interbloqueo

Un primer método consiste en preparar procedimientos de detección, como por ejemplo algún algoritmo *ad hoc* que se ejecute periódicamente auscultando al sistema. La frecuencia dependerá del sistema, de su complejidad y de las posibilidades de que se presente el interbloqueo.

La otra manera, denominada en muchas ocasiones reanudación, de eliminar el interbloqueo es forzar a que algunos de los procesos que lo han ocasionado se detengan en su ejecución y retrocedan en su ejecución hasta un momento en el que ya no presenten problemas con otros procesos por los recursos del sistema. Para hacer esto existen dos caminos: el más radical consiste en eliminar todos los procesos implicados; el más suave supone ir eliminando procesos implicados hasta que, llegado a un punto de la eliminación, el interbloqueo desaparezca.

IV. ADMINISTRACIÓN DE LA MEMORIA

Un programa generalmente reside en un disco como archivo binario ejecutable: para que sea ejecutado el programa debe ser llevado a la memoria y colocado dentro de un proceso. Dependiendo del tipo de administración de memoria en uso, el proceso puede transferirse entre el disco y la memoria durante su ejecución. El conjunto de proceso que están en el disco y esperando ser llevados a la memoria para su ejecución forma la cola de entrada.

El procedimiento normal consiste en seleccionar uno de los procesos de la cola de entrada y cargarlo en la memoria. A medida que el proceso se ejecuta, accede a las instrucciones y datos de la memoria. Con el tiempo, el proceso termina, y su espacio de memoria se declara como disponible.

Un compilador normalmente vinculará estas direcciones simbólicas con direcciones relocalizables. El editor de enlace o cargador, a su vez, vinculará las direcciones relocalizables con direcciones absolutas. Cada vinculación de instrucciones en otro.

De manera clásica, la vinculación de instrucciones y datos con direcciones de memoria puede hacerse en cualquier paso durante el recorrido:

Tiempo de compilación

Si se conoce al momento de la compilación en dónde residirá el proceso en la memoria, entonces se puede generar un código absoluto. Por ejemplo, si se sabe por adelantado que un proceso de usuario residirá a partir de la localidad R, entonces el código generado por el compilador comenzará en dicha localizad y se extenderá a partir de ahí hacia arriba. Si en un momento posterior, cambia la localidad de inicio, entonces será necesario recompilar este código. Los programas de MS-Dos con formato .com son vinculados con código absoluto en el momento de la compilación

Tiempo de carga.

Si no se conoce al momento de la compilación en dónde residirá el proceso en la memoria, entonces el compilador debe generar un código relocalizable. En este caso, la vinculación final se demora hasta el momento de la carga. So cambia la dirección de inicio, sólo se necesita recargar el código de usuario para incorporar este valor modificado.

Tiempo de ejecución

Si el proceso puede moverse durante su ejecución de un segmento de memoria a otro, entonces la vinculación debe ser demorada hasta el momento de la ejecución. Para que funcione este esquema, debe contarse con hardware especia. La mayoría de los sistemas operativo de propósito general utilizan este método

Espacio de direcciones lógicas contra físicas.

Una dirección generada por al cpu se conocen comúnmente como dirección lógica, en tanto que una dirección vista por la unidad de memoria, es decir, la que se carga en la memoria en el registro de direcciones de memoria se conoce como dirección física.

Sin embargo el esquema de vinculación de direcciones en el tiempo de ejecución da por resultado un ambiente en que las direcciones lógicas y físicas difieren. En este caso, generalmente nos referimos a la dirección lógica como dirección virtual. El conjunto de todas las direcciones lógicas es el espacio de direcciones físicas. Por lo tanto en el esquema de vinculación de direcciones en el tiempo de ejecución, los espacios de direcciones lógicas y físicas difieren

El mapeo en tiempo de ejecución de direcciones virtuales en direcciones físicas e realizado por la unidad de administración de memora (MMU), que es un dispositivo de hardware, este método requiere un soporte de hardware ligeramente diferente de la configuración. El registro base ahora se llama registro de relocalización. El valor en este registro se agrega a cada dirección generada por un proceso de usuario en el momento en que se envía a la memoria. Por ejemplo, si la base se encuentra en 14000; un acceso a la localidad 346 se mapea en la localidad 14, 346.

El programa de usuario proporciona direcciones lógicas; estas direcciones lógicas deben mapearse en direcciones físicas antes de ser utilizadas. El concepto de un espacio de direcciones lógicas que está vinculado a un espacio distinto de direcciones físicas es centrar para una correcta administración de la memoria.

Carga dinámica.

Todo programa y los datos de un proceso deben estar en la memoria física para la ejecución de éste. El tamaño de un proceso está limitado por el tamaño de la memoria física. Para conseguir una mejor utilización del espacio de memoria, podemos emplear la carga dinámica. Con la carga dinámica, una rutina no se carga si no hasta que es llamada. Todas las rutinas se mantienen en el disco en un formato de carga relocalizable. El programa principal se carga en la memoria y se ejecuta. Cuando una rutina necesita llamar a otra rutina, la rutina que hace el llamado verifica primeramente si la otra ha sido cargada. Si no ha sido cargada, el cargador de enlace relocalizables es llamado para colocar en memoria la rutina deseada y actualizar las tablas de direcciones del programa para reflejar este cambio.

Enlace dinámico y bibliotecas compartidas.

Algunos sistemas operativos solo soportan el enlace estático, en que las bibliotecas del lenguaje del sistema son tratadas como cualquier otro modulo objeto y son combinadas por el cargador en la imagen del programa binario. El concepto de enlace dinámico es similar al de la carga dinámica. En lugar de posponer la carga hasta el tiempo de ejecución, el enlace es el que se pospone. Esta característica se utiliza generalmente con bibliotecas del sistema, como las bibliotecas de subrutinas de lenguaje. Sin este servicio, todos los programas de un sistema necesitan tener una copia de su biblioteca de lenguaje (o por lo menos las rutinas a las que hace referencia el programa) incluida en la imagen ejecutable. Este requerimiento desperdicia espacio en disco así como memoria principal. Con el enlace dinámico, se incluye un Stub en la imagen por cada referencia a la rutina de biblioteca apropiada residente en memoria, o cómo cargar la biblioteca si la rutina no está ya presente.

A diferencia de la carga dinámica, el enlace dinámico generalmente requiere de ayuda del sistema operativo. Si los procesos en memoria están protegidos entre ellos, entonces el sistema operativo es la única entidad que puede verificar si la rutina que se necesita se encuentra en otro espacio de memoria del proceso, o que puede permitir que múltiples proceso accedan a las mismas direcciones de memoria

Superposiciones (overlays)

Para que un proceso pueda ser mayor que la cantidad de memoria asignada a él, podemos usar superposiciones. La idea de las superposiciones consiste en mantener en memora solo aquellas instrucciones y datos que se necesitan en un momento dado.

Como el caso de la carga dinámica, las superposiciones no requieren de algún soporte especial del sistema operativo. Pueden ser implementadas completamente por el usuario con estructuras de archivos sencillas, leyendo de los archivos a la memoria y luego saltando a dicha memoria y ejecutando las instrucciones recientemente leídas.

Intercambio (swapping)

Un proceso necesita estar en memoria para ser ejecutado. Sin embargo , un procese puede intercambiarse temporalmente de la memora a un almacén de respaldo, y luego llevarse de regresa la memoria para continuar su ejecución.

Una variante de esta política de intercambio se emplea para los algoritmos de planificación con base en prioridades. Si llega un proceso de mayor prioridad y quiere servicio, el administrador de la memoria puede intercambiar fuera de la memoria el proceso de menor prioridad de manera que pueda cargar y ejecutar el proceso de mayor prioridad. Cuando termina el proceso de mayor prioridad, el proceso de menor prioridad puede ser intercambiado de regreso a la memoria y continuar. Esta variante del intercambio también se conoce como desenrollar

Normalmente, un proceso que es intercambiado fuera de la memora será intercambiado de regreso sobre el mismo espacio que ocupaba al inicio. Esta restricción está dictada por el método de vinculación de direcciones., Si la vinculación se realiza en el momento del ensamble o de carga, entonces el proceso no puede ser movido a las localidades diferentes, si se esta empleando la vinculación en el tiempo de ejecución, entonces es posible intercambiar un proceso a un espacio de memoria diferente

El intercambio requiere un almacén de respaldo. Este almacén comúnmente un disco rápido. Debe ser lo suficientemente grande para acomodar copias de todas las imágenes de memoria para todos los usuarios, y debe proporcionar un acceso directo a estas imágenes. El sistema mantiene una cola de proceso listos compuesta por todos los proceso cuyas imágenes de memora están en el almacén de respaldo o en la memora y están listos para ejecutar. Siempre que el planificador de la cpu decide ejecutar un proceso, llama la despachador, este verifica si el siguiente proceso de la cola está en memora. Si el proceso no esta en memoria y no hay una región de memoria libre, el despachador intercambia hacia fuera un proceso que actualmente esté en memora e intercambia hacia adentro el proceso deseado. Luego, carga nuevamente los registros de manera normal y transfiere el control al proceso seleccionado.

Existen restricciones en el intercambio.

Nunca intercambiar un proceso con operaciones de e/s pendientes
Ejecutar operaciones de e/s solo con buffers del sistema operativo.

Asignación de memoria contigua.

Uno de los métodos más sencillos para la asignación de memoria consiste en dividirla en un número de particiones de tamaño fijo. Cada partición puede contener exactamente un proceso. De esta forma, el grado de multiprogramación está limitado por el número de particiones. Cuando una partición está libre, se selecciona un proceso de la cola de entrada y se carga en ella. Cuando el proceso termina, la partición queda disponible para otro proceso. Este método fue empleado originalmente por el sistema operativo os/36' de IBM, en la actualidad ya no se usa

El sistema operativo mantiene una tabla que indica que partes de la memoria están disponibles y cuales están ocupadas. Inicialmente toda la memoria está disponible para cuando llega un proceso y necesita memoria, buscamos un hueco suficientemente grande para este proceso. Si encontramos alguno, asignamos solo tanta memoria como se necesite manteniendo el resto disponible para satisfacer solicitudes futuras.

A medida que entran procesos al sistema, son colocados en una cola de entrada, el sistema operativo toma en cuenta los requerimientos de memoria y de cada proceso y la cantidad de espacio disponibles para determinar a cuales proceso se les asigna memora, cuando se asigna espacio a un proceso, este se carga en la memora y puede entonces competir por la cpu, cuando

un proceso termina, libera su memoria, misma que el sistema operativo llena con otro proceso de la cola de entrada.

En cualquier momento dado, tenerlos una lista de tamaños de bloque disponibles y la cola de entrada. El sistema operativo puede ordenar la cola de entrada de acuerdo con un algoritmo de planificación. La memoria se asigna a los procesos hasta que, finalmente, los requerimientos de memoria del siguiente proceso no puedan ser satisfechos; ningún bloque de memora (hueco) disponible es suficientemente grande para contener dicho proceso. El sistema operativo entonces espera hasta que esté disponible un bloque suficientemente grande, o salta hacia debajo de la cola de entrada para ver si satisfacen los requerimientos más pequeños de memora de algún otro proceso.

En general, en algún momento ha un conjunto de huecos, de varios tamaños, dispersos en toda la memoria. Cuando llega un proceso y necesita memoria, buscamos en este conjunto un hueco que sea suficientemente grande para el proceso. Si el hueco es demasiado grande, se divide en dos: una parte se asigna al proceso que está llegando y la otra se devuelve al conjunto de huecos. Cuando un proceso termina, libera su bloque de memoria, que es colocado de regreso en el conjunto de huecos. Si el nuevo hueco es adyacente a los otros huecos, combinados estos dos huecos adyacentes para formar un hueco más grande. En este punto, tal vez necesitemos verificar si hay procesos esperando memoria y su la memora recientemente liberada y recombinada puede satisfacer las demandas de algunos de estos procesos que están esperando

Este procedimiento es un caso particular del problema general de asignación dinámica de almacenamiento, que consiste en cómo satisfacer una solicitud de tamaño n a partir de una lista de huecos libres. Existen muchas soluciones a este problema. Se hace una búsqueda en conjunto de huecos para determinar cual s el mejor huevo para asignar. Las estrategias del primer ajuste, el mejor ajuste y el peor ajuste son las que se empelan más comúnmente para seleccionar un hueco libre del conjunto de huecos disponibles.

Primer ajuste
Asigne EL primer hueco que sea suficientemente grande. La búsqueda puede comenzar ya sea al principio del conjunto de huecos o donde termino la búsqueda previa del primer ajuste. Podemos dejar de buscar tan pronto como encontremos un hueco libre que sea suficientemente grande.

Mejor ajuste
Asigne el hueco más pequeño que sea suficientemente grande. Debemos hacer una búsqueda en toda la lista, a menos que la lista se mantenga ordenada por tamaño. Esta estrategia produce el hueco más pequeño restante.

Peor ajuste.
Asigne el hueco más grande. Nuevamente, debemos hacer una búsqueda en toda la lista, a menos que esté ordenada por tamaño. Esta estrategia produce el hueco más grande que queda, el cual puede ser más útil que el hueco más pequeño que queda en el caso del enfoque de mejor ajuste.

Los algoritmos que se acaban de presentar adolecen de una fragmentación externa a medida que los procesos se cargan y remueven de la memoria, el espacio de memoria libre se divide en pequeñas piezas. La desfragmentación externa ocurre cuando existe suficiente espacio de memoria total para satisfacer una solicitud, pero no es contigua; el almacenamiento se fragmenta en una gran número de huecos pequeños

Paginación

La paginación es un esquema que permite que el espacio de direcciones lógicas de un proceso no sea contiguo. La paginación evita el problema mayor de ajustar las porciones de memoria de tamaño variable en el almacén respaldo, del cual sufrirían la mayoría de los esquemas anteriores de administración de la memoria.

Método básico.

La memoria física se descompone en bloques de tamaño fijo denominados marcos. La memoria lógica también se descompone en bloques del mismo tamaño denominados páginas. Cuando se va ejecutar u proceso, sus páginas se cargan desde el almacén de respaldo en cualquier marco de memoria disponible. El almacén de respaldo se divide en bloques de tamaño fijo que son del mismo tamaño que los marcos de la memoria.
Cada dirección generada por la cpu se divide en dos partes: un número de página (p) y un desplazamiento de página (d). El número de página se emplea como un índice en una tabla de páginas. La tabla de páginas contiene la dirección base de cada página en la memoria física. Esta dirección base se combina con el desplazamiento de página para definir la dirección física de la memoria que se envía a la unidad de memoria.

El tamaño de página (así como el tamaño del marco) es definido por el hardware. El tamaño de una página es normalmente una potencia de 2 y varía entre 512 bytes y 16 megabytes por página, dependiendo de la arquitectura de la computadora. La selección de una potencia de 2 como tamaño de página hace que la traducción de una dirección lógica a un número de desplazamiento de página sea particularmente fácil. Si el tamaño del espacio de direcciones lógicas es 2 m y el tamaño de una página es 2 n unidades de direccionamiento (bytes o palabras), entonces los m-n bits de orden elevado de una dirección lógica designan el número de página, y los n bits de orden bajo designan el desplazamiento de página, en donde p es un índice en la tabla de páginas y d es el desplazamiento dentro de la página.

Observe que la paginación misma es una forma de relocalización dinámica. El hardware de paginación vincula cada dirección lógica con alguna dirección física. El lector observadores habrá dado cuenta que el empleo de la paginación es similar al uso de una tabla de registros base (de relocalización)

Un aspecto importante de la paginación es la distinción clara entre la visión que tiene el usuario de la memoria y la memoria física real: el programa de usuario ve a dicha memoria como un solo espacio contiguo, La diferencia entre la visión que el usuario tiene de la memoria y la memoria física real se reconcilia mediante el hardware de traducción de direcciones, las direcciones lógicas se traducen a direcciones físicas. Este mapeo está oculto al usuario y es controlado por el sistema operativo.

Debido a que el sistema operativo está administrando la memoria física, debe estar consciente de los detalles de asignación de la misma: qué marcos se asignan, cuáles marcos están disponibles, cuántos marcos en total hay, etc. Esta información se mantiene generalmente en una estructura de datos denominada tabla de marcos. La tabla de marcos tiene una entrada por cada marco físico de página, indicado si el último está libre o si está asignado, a que página de qué proceso o proceso

Estructura de la tabla de páginas

Cada sistema operativo tiene sus propios métodos para almacenar tablas de páginas. La mayoría asigna una tabla de páginas por cada proceso. Un apuntador a la tabla de páginas se almacena con los demás valores de registros (como el contador de instrucciones) en el bloque de control del proceso. Cuando se le dice al despachador que inicie un proceso, debe recargar los registros del usuario y definir los valores correctos de la tabla de páginas de hardware a partir de la tabla de páginas del usuario que está almacenada.

Soporte de hardware.

La implementación en hardware de la tabla de páginas puede hacerse de varias formas. En el caso más sencillo, la tabla se implementa como un conjunto de registros dedicados. Estos registros deberán construirse con una lógica de muy alta velocidad para que la traducción de direcciones de página sea eficiente. Todo acceso a la memoria debe pasar por el mapa de paginación, por lo que

la eficiencia es un aspecto importante. El despachador de la CPU recarga estos registros, al igual que recarga a los demás registros. Las instrucciones para cargar o modificar los registros de la tabla de páginas son, por supuesto, privilegiadas.

El uso de registros para la tabla de páginas es satisfactorio si la tabla es razonablemente pequeña. La mayoría de las computadoras actuales, sin embargo permiten que la tabla de páginas sea muy grandes un millón por ejemplo, el empleo de registros rápidos para implementar la tabla de páginas no es factible. Más bien dicha tabla se mantiene en la memoria principal y un registro base de tabla de páginas.

La solución estándar a este problema es usar una cache de hardware especial, pequeña y de búsqueda rápida, conocida también como registros asociativos o buffers de traducción de vista lateral. Se construye un conjunto de registros asociativos de memoria de alta velocidad. Cada registro consta de dos partes: una llave un valor cuando los registros asociativos se presentan con un elemento, éste se compara con todas las llaves simultáneamente, si se encuentra el elemento, se produce como salida el campo de las llave del valor correspondiente, los registros asociativos se emplean con tablas de páginas de la siguiente manera, estos asociativos sólo contienen unas cuantas entradas de la tabla de páginas. Cuando la cpu genera una dirección lógica, su número de página se presenta a un conjunto de registros asociativos que contienen números de páginas y sus correspondientes números de marco.

Protección

La protección de la memora en un ambiente con paginación se realiza mediante bits de protección que están asociados con cada marco. Estos bits normalmente se mantienen en la tabla de páginas. Un bit puede definir que una página sea de lectura y escritura, o sólo de lectura. Cada referencia a la memoria pasa por la tabla de páginas para encontrar el número correcto de marco, generalmente se agrega un bit más a cada entrada de la tabla de páginas: Un bit de validez-invalidez

Paginación con niveles múltiples

La mayoría de los sistemas de computo modernos soportan un espacio grande de direcciones lógicas (2 32 / 2 64) En tales ambientes, la tabla de páginas misma se vuelve excesivamente grande. Una solución sencilla a este problema consiste en dividir la tabla de páginas en piezas más pequeñas. Hay varias formas de lograr esta división.

Una forma consiste en utilizar un algoritmo de paginación con dos niveles, en que la tabla de páginas misma también se pagina,

Tabla de páginas invertidas

Cada proceso tiene generalmente una tabla de páginas asociadas con él. La tabla de páginas tiene una entrada por cada página que el proceso está utilizando (o una ranura por cada dirección virtual, independientemente de la validez de esta última) . Esta es una representación natural de la tabla ya que los procesos hacen referencia las páginas mediante las direcciones virtuales de las mismas. El sistema operativo debe traducir después esta referencia a una dirección física de memoria. Debido a que la tabla está ordenada por dirección virtual, el sistema operativo puede calcular en qué lugar de la tabla está la entrada de la dirección física asociada, y usar dicho valor directamente. Una de las desventajas de este método es que cada tabla de páginas puede constar de millones de entradas. Estas tablas consumen grandes cantidades de memoria física únicamente para llevar un registro de cómo se está utilizando la otra memoria física.

Para resolver este problema se puede usar una tabla de páginas invertida. Una tabla de páginas invertida tiene una entrada por cada página (marco) real de memoria. Cada entrada consiste en la dirección virtual de la página almacenada en dicha ubicación de memora real, con información

acerca del proceso que posee dicha página. Así , sólo hay una tabla de páginas en el sistema, y sólo tiene una entrada por cada página de la memoria física..
Cada entrada de la tabla de páginas invertida es un par < id-proceso, número- de- página>, cuando ocurre una referencia a la memoria, parte de la dirección virtual, que consta de <id-proceso, número – de- página>, se presenta al subsistema de la memoria. Luego, en la tabla de páginas invertida se busca una correspondencia. Si se encuentra alguna- por ejemplo, en la entrada i- entonces se genera la dirección física <i-desplazamiento>. Si no se encuentra una correspondencia, es que se ha intentado un acceso ilegal a una dirección.

Aunque este esquema disminuye la cantidad de memoria necesaria para almacenar cada tabla de páginas, incrementa la cantidad de tiempo necesario para buscar en la tabla cuando ocurre una referencia a una página. Como la tabla de páginas invertida está ordenada por dirección física, pero las búsquedas ocurren en direcciones virtuales, podría ser necesario buscar aliviar este problema, se usa una tabla hash para limitar la búsqueda a una entrada – o a lo sumo unas cuantas entradas- de la tabla de páginas. Por supuesto, cada acceso a la tabla hash agrega al procedimiento una referencia a memoria, por lo que una referencia a la memoria virtual requiere de por lo menos dos lecturas a la memoria real: una para la entrada en la tabla hash y una para la tabla de páginas. Para mejorar el rendimiento, utilizamos registros asociativos de memoria para retener las entradas localizadas recientemente. Primero se busca en estos registros, antes de consultar la tabla hash

Páginas compartidas.

Otra venta de la paginación es la posibilidad de compartir un código común. Esta consideración es particularmente importante en un ambiente de tiempo compartido. Considere un sistema que soporta 40 usuarios, cada uno de los cuales ejecuta un editor de textos consta de 150 k y 50 k de espacio para datos, se necesitan 8000 l para soportar a los 40 usuarios. Sin embargo, si el código es reentrante, se puede compartir

El código reentrante (también denominado código puro) es un código que no puede modificarse a sí mismo. Si el código es reentrante, entonces nunca cambia durante la ejecución. Por lo tanto, dos o más procesos pueden ejecutar el mismo código al mismo tiempo. Cada proceso tiene su propia copia de registros y almacenamiento de datos para contener los datos para la ejecución de los proceso

Segmentación

La segmentación es un esquema de administración de la memoria que soporta esta visión del usuario. Un espacio de direcciones lógicas es un conjunto de segmentos. Cada segmento tiene un nombre y una longitud. Las direcciones especifican tanto el nombre del segmento como el desplazamiento dentro del mismo. Por lo tanto, el usuario especifica cada dirección mediante dos cantidades. Un nombre de segmento y un desplazamiento.

Protección y compartimiento.

Una ventaja particular de la segmentación es la asociación de la protección con los segmentos. Debido a que los segmentos representan una porción del programa definida semánticamente, es probable que todas las entradas del segmento se utilicen en la misma forma. Por lo tanto, hay algunos segmentos que son instrucciones, en tanto que otros son datos. En una arquitectura moderna, las instrucciones no pueden modificarse por ellas mismas, por lo que los segmentos de instrucciones se pueden definir como de sólo lectura o sólo ejecución.

Técnica de la memoria virtual

Está basada en la anterior. Trata de cargar en memoria sólo lo que sea estrictamente necesario para su funcionamiento correcto. Así, el sistema tiene que decidir cuándo cargar las secciones del

programa -de forma anticipada o sirviendo a las solicitudes-, dónde cargar cada sección -en el primero disponible o en el segmento de memoria que se revele como el más adecuado-, y si existe reemplazamiento -es decir, si se sustituye alguna porción de memoria por una sección nueva. utilizando algún algoritmo implementado al efecto-.

Según la primera de las opciones, el sistema cargará una página cuando ésta sea llamada siguiendo un procedimiento similar a una técnica *swapping* o de intercambio de memoria. Así, al solicitarse una página determinada, el sistema operativo consulta la tabla de páginas y al localizarla la marca con un bit de válido y carga la parte correspondiente del programa en memoria. También debe ser capaz el sistema de solucionar los casos en que se haga referencia a una página que no está cargada en memoria, o bien cuando el hardware detecta un error de dirección, tratado por el sistema operativo como si la página no existiera.

El reemplazamiento es un método útil para cuando se dan situaciones de sobreasignación de memoria, es decir, cuando ésta está saturada, con lo que aparecen referencias a páginas que no están cargadas en memoria. Para solucionarlo el sistema operativo saca de la memoria la página menos usada -página víctima- y la carga en una memoria más lenta, por ejemplo en disco guardando lo que el programa haya escrito en memoria mientras tanto. Después actualiza la tabla de páginas, puesto que habrá un elemento menos. Después sustituye a la víctima cargando la nueva página en memoria. Y por último vuelve a actualizar la tabla de páginas pues habrá un elemento nuevo.

Parece de sentido común que si un programa necesita, para su ejecución, cargar varias páginas en memoria, y alguna de ellas debe ser sustituida, es conveniente elegir a la que se hará referencia por el sistema con mayor posterioridad.

Los algoritmos más usados para gestionar el ir y venir de las páginas a la memoria para asignarlas a los diferentes marcos, son:

- FIFO primera en llegar a la memoria es la primera en ser considerada víctima, es decir, aplicando un criterio de antigüedad en la memoria. Sin embargo, aunque es barato y fácil, es poco eficiente, pues en muchas ocasiones provoca demoras
- LRU (*Less Recently Used*): Es decir, los menos usados últimamente. O sea, considerará una buena víctima a aquella página que hace tiempo que no se usa. Para ello se usan soluciones hardware -aplicar registros contadores de tiempo- o software -pilas de números de página (la página que se usa se pone a la cabeza de la pila)-.
- LFU (*Less Frecuency Used*): La página menos usada, comparando, por ejemplo, los contadores de uso asociados a cada página.
- SO (*Second oportunity*): Busca la página que más tiempo lleva en memoria y si está marcada como usada -por ejemplo, con un bit 1-, la remarca con un 0 y le da una segunda oportunidad.
- Localidad: sólo busca páginas víctimas entre las vecinas a las páginas que el programa está utilizando. Esto tiene como objetivo evitar que crezca el número de páginas asignadas a un proceso.
- Por el conjunto del trabajo: El sistema operativo calcula cuánta memoria necesitará para los procesos que constituyen un programa -o los módulos de un proceso- de forma que si estima que no tiene memoria suficiente para todos, no lo carga en la memoria. Si sí tiene, entonces sí lo carga. Para ello el sistema operativo “observa” cada cierto tiempo las N referencias realizadas la última vez para saber cuántas páginas y marcos están en uso.
- Frecuencia de faltas de página: El sistema operativo calcula cuántas faltas de página se han dado en el último intervalo de tiempo. Si se han dado pocas es que sobre memoria, si es al revés, es que falta. Si se pasa del máximo, entonces el sistema operativo asigna más marcos de memoria.

V. ADMINISTRACIÓN DE ARCHIVOS

Concepto de archivo

Las computadoras pueden almacenar información en diversos medios de almacenamiento (discos magnéticos, cintas magnéticas y discos ópticos). Para facilitar el uso del sistema de cómputo, el sistema operativo proporciona una cisión lógica y uniforme de almacenamiento de la información. Este sistema hace una abstracción de las propiedades físicas de sus dispositivos de almacenamiento para definir una unidad lógica de almacenamiento, el archivo.

El sistema operativo mapea a los archivos sobre dispositivos físicos. Estos dispositivos de almacenamiento general son no volátiles, por lo que sus contenidos persisten aun durante fallas de energía y rearranques del sistema.

Un archivo es una colección de información relacionada, con un nombre asignado, que se graba en almacenamiento secundario. Desde la perspectiva de un usuario, un archivo es la porción más pequeña de almacenamiento secundario lógico, es decir no pueden escribirse datos en almacenamiento secundario, a menos que se encuentren dentro de un archivo. Comúnmente los archivos representan programas y datos. Los archivos de datos pueden ser numéricos, alfabéticos, alfanuméricos o binarios. Los archivos pueden ser de forma libre o pueden tener un formato rígido. En general, un archivo es una secuencia de bits, bytes, líneas o registros cuyo significado es definido por el creador y el usuario del archivo.

Atributos de un archivo

Un archivo tiene entre otros atributos que varían de un sistema operativo a otro, pero que típicamente son:

- Nombre: el nombre simbólico del archivo es la única información que se mantiene en forma legible para los humanos.
- Tipo: esta información es necesaria para aquellos sistemas que soportan diferentes tipos.
- Ubicación: esta información es un apuntador a un dispositivo y a la ubicación del archivo en dicho dispositivo.
- Tamaño: en este atributo se incluyen el tamaño actual del archivo (en bytes, palabras o bloques) y posiblemente el tamaño permitido.
- Protección: información de control de acceso que determina quién puede leer, escribir, ejecutar, etc., el archivo.
- Hora, fecha e identificación del usuario: esta información puede mantenerse para la creación, última modificación y el último uso. Estos datos pueden ser útiles para protección, seguridad y control de uso.

Operaciones sobre los archivos

El sistema operativo proporciona llamadas al sistema para crear, escribir, leer , reposicionar, borrar y truncar archivos.

Para crear un archivo se requieren dos pasos, primero se debe encontrar espacio para éste en el sistema de archivos, en segundo lugar se debe hacer una entrada en el directorio para el nuevo archivo. La entrada en el directorio registra el nombre del archivo y su ubicación en el sistema de archivos.

Para escribir un archivo hacemos una llamada al sistema especificando tanto el nombre del archivo como la información que se va a escribir en él. Dado el nombre del archivo, el sistema busca en el directorio para encontrarle una ubicación. El sistema debe mantener un apuntador de escritura a la ubicación del archivo donde va a tener lugar la siguiente escritura. El apuntador de escritura debe actualizarse siempre que ocurre una escritura.

Para leer un archivo, empleamos una llamada al sistema que especifica el nombre del archivo y el lugar (en la memoria) donde deberá colocarse el siguiente bloque del mismo. Nuevamente se busca en el directorio la entrada asociada, y el sistema mantiene un apuntador de lectura a la ubicación en el archivo en donde va a tener lugar la siguiente lectura. Una vez que se ha realizado la operación, el apuntador de lectura se actualiza. Debido a que en general un archivo se está leyendo o escribiendo, la mayoría de los sistemas mantienen sólo un apuntador de la posición actual en el archivo. Tanto la operación de lectura como la de escritura emplean este mismo apuntador, ahorrando espacio y reduciendo la complejidad del sistema.

Reposicionarse dentro de un archivo, primero se busca en el directorio la entrada apropiada, y se asigna un valor dado a la posición actual del archivo. El reposicionamiento dentro de un archivo no necesita incluir una operación real de E/S. Esta operación sobre el archivo también se conoce como búsqueda en archivo.

Para borrar un archivo, buscamos en el directorio el archivo designado. Una vez que se ha encontrado la entrada asociada, liberamos todo el espacio del archivo (para que pueda ser reutilizado por otros archivos) y borramos la entrada al directorio.

Truncar un archivo, hay ocasiones en que el usuario desea que los atributos de un archivo permanezcan iguales, pero quiere borrar el contenido del archivo. En lugar de obligar al usuario a borrar el archivo y después volver a crearlo, esta función permite que todos los atributos permanezcan sin modificación (excepto la longitud del archivo), pero restableciendo el archivo a una longitud cero.

Tipos de archivos

En el diseño de un sistema de archivos, y de todo el sistema operativo, es si el sistema operativo deberá reconocer y soportar tipos de archivos. Si un sistema operativo reconoce le tipo de un archivo, entonces puede operar sobre el archivo en formas razonables.
Una técnica común para implementar tipos de archivos consiste en incluir el tipo como parte del nombre del archivo. El nombre se divide en dos partes, el nombre y la extensión, separados por un punto. De esta forma el usuario y el sistema operativo pueden saber con el nombre que tipo de archivo es. En MS-DOS la extensión se utiliza para indicar el tipo de archivo y el tipo de operaciones que pueden realizarse sobre dicho archivo.

Tipo de archivo	Extensión usual	Función
Ejecutable	exe, com, bin o ninguna	Programa en lenguaje de máquina listo para correr
Objeto	obj, o	Compilado, en lenguaje de máquina, no enlazado
Código fuente	c, cc, pas, java, asm, a	Código fuente en varios lenguajes
Por lotes	bat, sh	Comandos al intérprete de comandos
Texto	txt, doc	Datos textuales, documentos
Procesador de palabras	wpd, tex, doc, etc.	Variso formatos de procesador de palabras
Biblioteca	lib, a, DLL	Bibliotecas de rutinas para programadores
Impresión o vista	ps, dvi, gif	Archivo ASCII o binario en un formato para impresión o vista
Archivos	arc, zip, tar	Archivos relacionados agrupados en un archivo, a veces comprimido, para archivarlo o almacenarlo

Métodos de acceso

Cuando se utiliza la información almacenada en un archivo, se debe acceder a ella y llevarla a la computadora y para hacerlo existen varias formas de acceder a la información de un archivo.

El acceso secuencial es el método de acceso más sencillo, la información en el archivo se procesa en orden, un registro después de otro, por lo mismo es el método de acceso más común.
El acceso directo o relativo, un archivo está compuesto de registros lógicos de longitud fija que permiten a los programas leer y escribir registros rápidamente sin un orden particular. Este método se basa en un modelo de archivo en disco, el cual permite un acceso aleatorio a cualquier bloque de archivo. En le caso del acceso directo, el archivo es visto como una secuencia numerada de bloques o registros. Un archivo de acceso directo permite leer o escribir bloques arbitrariamente.

Es posible construir otros métodos de acceso sobre un método de acceso directo. Estos métodos adicionales generalmente implican la construcción de un índice para el archivo. El índice contiene apuntadores a los diversos bloques. Para encontrar un registro en un archivo, primero buscamos en el índice y luego usamos el apuntador para acceder directamente al archivo y encontrar el registro deseado.

Estructura de los directorios

Los sistemas de archivos de las computadoras pueden ser extensos, por lo que se necesita organizarlos de alguna manera,. Dicha organización se realiza en dos partes, primero el sistema de archivos se descompone en particiones o volúmenes y en segundo lugar cada partición contiene información acerca de los archivos dentro de ella. Esta información se mantiene en entradas en un directorio del dispositivo o tabla de contenido del volumen.
Existen varios esquemas para definir la estructura lógica de un directorio, a continuación describiremos los más comunes.

El directorio de un solo nivel, que es la estructura más sencilla, donde todos los archivos están contenidos en el mismo directorio, el cuál es fácil de soportar y entender.
El directorio de dos niveles, consiste en crear para cada usuario su propio directorio de archivos de usuario, el cual tiene una estructura similar, pero lista solo los archivos de un usuario.
Directorios con estructura de árbol, es similar al de dos niveles, un directorio contiene un conjunto de archivos o subdirectorios y así mismo cualquiera de esos subdirectorios tiene más subdirectorios.
También podemos mencionar el directorio de gráfica acíclcia, es una gráfica sin ciclos, permite que los directorios tengan subdirectorios y archivos compartidos.

Protección de archivos

La información que se guarda en un archivo, debe de ser protegida contra un acceso incorrecto, por lo que es necesario un acceso controlado. Los mecanismos de protección proporcionan un acceso controlado limitando los tipos de acceso que pueden hacerse a los archivos. El acceso se permite o se niega dependiendo de varios factores, uno de los cuales es el tipo de acceso solicitado. Se pueden controlar varios tipos distintos de operaciones:

Leer: leer un archivo
Escribir: escribir o volver a escribir el archivo
Ejecutar: cargar el archivo en memoria y ejecutarlo
Anexar: escribir nueva información al final del archivo
Borrar: borrar el archivo y liberar espacio para una posible reutilización
Listar: listar el nombre y los atributos del archivo

La protección consiste en hacer que el acceso dependa de la identidad del usuario. Varios usuarios pueden necesitar diferentes tipos de acceso a un archivo o directorio. El esquema más general para implementar un acceso que dependa de la identidad consiste en asociar una lista de acceso

con cada archivo y directorio, especificando para cada usuario de la lista el nombre y los tipos de accesos permitidos. Cuando un usuario solicita acceso a un archivo particular, el sistema operativo verifica la lista para el acceso solicitado, este se permite. En caso contrario, ocurre una violación de protección y se le niega el acceso al archivo al trabajo del usuario.

Estructura del sistema de archivos

Los discos proporcionan la mayor parte del almacenamiento secundario en el cual se mantiene un sistema de archivos. Para mejorar la eficiencia de las operaciones E/S, realizamos las transferencias entre la memoria y el disco en unidades de bloques.

Organización del sistema de archivos

Para proporcionar un acceso eficiente y conveniente al disco, el sistema operativo impone un sistema de archivos que permite almacenar, localizar y recuperar datos fácilmente. El desarrollo de un sistema de archivos plantea dos problemas distintos de diseño. El primer problema es definir cómo deberá ser la apariencia del sistema de archivos para el usuario, lo que comprende la definición de un archivo y sus atributos, las operaciones permitidas sobre el archivo, y la estructura de directorios para organizar los archivos. A continuación se deben crear algoritmos y estructuras de datos que mapeen el sistema de archivos lógico en los dispositivos físicos de almacenamiento secundario. El sistema de archivos mismo generalmente está compuesto por varios niveles distintos. Cada nivel en el diseño emplea las características que utilizarán los niveles superiores.

Métodos de asignación

Generalmente los archivos se almacenan en un mismo disco, el principal problema es como asignar espacio a estos archivos de manera que el espacio libre en el disco se utilice de manera eficaz y se pueda tener acceso a ellos con rapidez. Existen tres métodos principales de asignación de espacio de disco y que se utilizan ampliamente:

Contigua: requiere que cada archivo ocupe un conjunto de bloques contiguos en el disco. Las direcciones del disco definen un ordenamiento lineal en el mismo.
Enlazada: cada archivo es una lista enlazada de bloques de disco, los bloques pueden estar dispersos en cualquier parte del disco. El directorio contiene un apuntador al primer y último bloque del archivo.
Indizada: lleva todos los apuntadores juntos a una ubicación, llamada el bloque índice. Cada archivo tiene su propio índice, que es un arreglo de direcciones de bloques en disco.

Administración del espacio libre

Debido a que sólo existe una cantidad limitada de espacio en disco, es necesario reutilizar para nuevos archivos el espacio dejado por los archivos eliminados, si es posible. Para dar seguimiento al espacio libre en disco, el sistema mantiene una lista de espacio libre, la cual registra todos los bloques del disco que están libres (los que no están asignados a algún archivo o directorio). Para crear un archivo, buscamos en la lista de espacio libre, la cantidad de espacio requerido y asignamos dicho espacio al nuevo archivo. Este espacio se remueve después de la lista de espacio libre. Cuando se borra un archivo, su espacio en disco se agrega a la lista de espacio libre. La lista de espacio libre con frecuencia se implementa como un mapa de bits o vector de bits. Cada bloque se representa mediante un bit, si está libre el bit es1, si está ocupado el bit es 0.

Los métodos de asignación de espacio libre también influyen en la eficiencia del uso del espacio en disco, el desempeño del sistema de archivos y la confiabilidad del almacenamiento secundario. Los métodos utilizados incluyen vectores de bits y listas enlazadas. Las optimizaciones incluyen la agrupación, el conteo y la FAT, que coloca la lista enlazada en un área contigua.

Recuperación

Debido a que los archivos y directorios se mantienen tanto en la memoria principal como en el disco, debemos asegurarnos de que un fallo del sistema no dé por resultado la pérdida de datos o inconsistencia en los mismos.

El verificador de consistencia compara los datos en la estructura de directorios con los bloques de datos en el disco y trata de corregir las inconsistencias que encuentre. Los algoritmos de asignación y de administración de espacio libre determinan qué tipos de problemas puede encontrar el verificador y el grado del éxito que tendrá en la corrección de dichos problemas.

Para evitar que los datos de los archivos se pierdan para siempre, podemos emplear programas de sistema para respaldar datos del disco a otro dispositivo de almacenamiento.

La recuperación de la pérdida de un archivo individual, o de todo un disco, puede entonces implicar simplemente restablecer los datos a partir de un respaldo.

VI. SEGURIDAD

La protección es cualquier mecanismo para controlar el acceso de programas, procesos o usuarios a los recursos definidos por el sistema de cómputo. Este mecanismo debe proporcionar un medio para la especificación de los controles que se van a imponer y otro para hacerlos cumplir. Distinguimos entre protección y Seguridad de, la cual es una media de confianza de que se conservar para la integridad de un sistema y de sus datos.

Metas de la protección

La protección se concibió originalmente como un adjunto a los sistemas operativos de multiprogramación, de manera que usuarios poco confiables pudieran compartir con seguridad un espacio común de nombres lógicos. Un directorio de archivos. O compartir un espacio común de nombres físicos. La memoria. Los conceptos de protección modernos han evolucionado para incrementar la confianza de cualquier sistema complejo que hace uso de recursos compartidos

Existen varias razones para proporcionar protección. La mas obvia es la necesidad de impedir la violación maliciosa e intencional de una restricción de acceso por parte de un usuario. Sin embargo, de importancia más general es la necesidad de asegurar que cada programa activo utilice recursos del sistema sólo en formas que correspondan a las políticas establecidas para el uso de tales recursos. Este requerimiento es indispensable para un sistema confiable.

La protección puede mejorar la confiabilidad si detecta errores latentes en las interfaces entre los subsistemas componentes. Una detección oportuna de los errores de interfaz a menudo puede impedir la contaminación de un subsistema saludable por un subsistema que está funcionando incorrectamente. Un recurso no protegido no puede defenderse contra el uso o el abuso.

El papel de la protección en un sistema de cómputo es proporcionar un mecanismo para hacer cumplir las políticas que rigen el uso de recursos. Estas políticas pueden fijarse de diversas formas. Algunas están establecidas desde el diseño del sistema, en tanto que otras son formuladas por la administración de un sistema. Otras más son definidas por los usuarios individuales para proteger sus propios archivos y programas. Un sistema de protección debe tener la flexibilidad para hacer cumplir una variedad de políticas que pueden establecerse. Por estas razones, la protección ya no puede considerarse solamente como una cuestión de intereses para el diseñador de un sistema operativo también deberá estar disponible como una herramienta para el programador de aplicaciones de modo que los recursos creados y soportados por un subsistema de aplicaciones pueda protegerse contra abusos., un principio importante es la separación entre política y mecanismo. Los mecanismos determinan cómo se hará algo. En contraste, las políticas deciden que se hará.

Dominio de la protección

A un proceso solo se le deben permitir acceder aquellos recursos a los que tiene acceso autorizado. Además, en un momento dado sólo debe ser capaz de acceder aquellos recursos que requiere en ese momento para completar su tarea. Este requerimiento, conocido comúnmente como el principio de necesidad de conocer, es útil para limitar la cantidad de daño que puede causar en el sistema un proceso que está fallando.

Estructura del dominio

Cada dominio define un conjunto de objetos y los tipos de operaciones que pueden invocarse sobre cada objeto, la habilidad para ejecutar una operación sobre un objeto es un derecho de acceso. Un dominio es una colección de derechos de acceso, cada uno de los cuales es un par ordenado <nombre-de objeto, conjunto-de- derechos>

Un dominio puede obtenerse en una variedad de formas:

Cada usuario puede ser un dominio. En este caso, el conjunto de objetos a los que se puede acceder depende de la identidad del usuario. La conmutación de dominios ocurre cuando se cambia el usuario- generalmente cuando un usuario se desconecta y otro se conecta.

Cada proceso puede se un dominio. En este caso, el conjunto de objetos a los que se puede acceder depende de la identidad del proceso. La conmutación de dominios corresponde a un proceso enviando un mensaje a otro proceso, y luego esperando una respuesta.

Cada procedimiento puede ser un dominio. En este caso, el conjunto de objetos que se puede acceder corresponde a las variables locales definidas dentro del procedimiento. La conmutación de dominios ocurre cuando se hace una llamada a un procedimiento.

Protección en Unix.

En el sistema operativo UNIX, un dominio está asociado con el usuario. La conmutación del dominio corresponde a cambiar temporalmente la identificación del usuario. Este cambio se logra mediante el sistema de archivos de la siguiente manera. Una identificación de propietario y un bit de domino (conocido como el bit seguid) están asociados con cada archivo. Cuando un usuario (con el ID de usuario = A) comienza a ejecutar un archivo que pertenece a B, y cuyo bit de dominio asociado está apagado, el ID de usuario del proceso se fija en A. cuando el bit seguid está prendido, el valor del ID de usuario se fija en el del propietario del archivo B cuando el proceso termina, finaliza este cambio temporal del ID de usuario.

PROTECCIÓN EN MULTICS

En el sistema MULTICS los dominios de protección están organizados jerárquicamente en una estructura de anillo. Cada anillo corresponde a un solo dominio

Matriz de acceso

El modelo de protección puede verse de manera abstracta como una matriz, denominada matriz de acceso. Las filas de la matriz de acceso representan dominios y las columnas representan objetos. Cada entrada en la matriz consiste en un conjunto de derechos de acceso, debido a que los objetos están definidos explícitamente por la columna, podemos omitir el nombre del objeto en el derecho de acceso.

Implementación de la matriz de acceso

Tabla global.
La tabla generalmente es grande y por lo tanto no puede mantenerse en memoria principal, por lo que se necesita e/s adicional. Lo que se emplea son las técnicas de memoria virtual para administrar esta tabla.

Listas de acceso para objetos.
Cada columna en la matriz de acceso puede implantarse como una lista de acceso para un objeto, obviamente, las entradas vacías pueden descartase la lista resultante para cada objeto consta de pares ordenados < dominio, conjunto de derechos>, que definen todos los dominios con un conjunto no vació de derechos de acceso para dicho objeto. Este enfoque puede extenderse fácilmente para definir una lista más un conjunto por omisión de derechos de acceso.

Listas de capacidades para dominios.
Una lista de capacidades para un dominio es una lista de objetos junto con las operaciones que se permiten sobre dichos objetos. Un objeto se representa mediante su nombre o dirección física, denominada capacidad.

Mecanismo de cerradura-llave
El esquema de cerradura-llave es un compromiso entre lista de acceso y listas de capacidades. Cada objeto tiene una lista de patrones de bits únicos, denominados cerraduras de manera similar, cada dominio tiene una lista de patrones de bits únicos denominados llaves.

Comparación
Las listas de capacidades no corresponden directamente a las necesidades de los usuarios; no obstante, son útiles para localizar información para un proceso particular, el proceso que intente hacer un acceso debe presentar una capacidad para dicho acceso luego, el sistema de protección solo necesita verificar que la capacidad sea validad. Sin embargo, la revocación de capacidades puede ser ineficiente.

Revocación de los derechos de acceso

En un sistema de protección dinámica, en ocasiones tal vez sea necesario revocar derechos de acceso a objetos que son compartidos por diferentes usuarios. Pueden surgir varias cuestiones acerca de la revocación.

Inmediata contra demorada
Selectiva contra general
Parcial contra total

La revocación es fácil con un esquema de lista de acceso. Se hace una búsqueda en la lista de acceso del o los derechos de acceso que van a revocar, y se eliminan de la lista. La revocación es inmediata, y pueden ser generales o selectivas, totales o parcial y permanentemente temporales.

Las capacidades, sin embargo, presenta un problema de revocación mucho más difícil debido a que las capacidades están distribuidas a lo largo del sistema, debemos encontrarlas antes de poder revocarlas. Existen diferentes esquemas para implementar la revocación de capacidades, incluyendo las siguientes.

Readquisición. Periódicamente se eliminan capacidades de cada dominio, si un proceso desea utilizar una capacidad, tal vez encuentre que dicha capacidad ha sido eliminada. El proceso trata entonces de readquirir la capacidad. Si el acceso ha sido revocado, el proceso no podrá readquirirla.

Apuntadores hacia atrás: Con cada objeto se mantiene una lista de apuntadores apuntando a todas las capacidades asociadas con dicho objeto. Cuando se requiere revocación, podemos seguir a estos apuntadores, cambiando las capacidades según sea necesario. Este esquema ha sido adoptado en el sistema MULTICS.

Indirección: Las capacidades no apuntan a los objetos directamente, sino que apuntan indirectamente. Cada capacidad apunta a una entrada única en una tabla global, la cual, a su vez, apunta al objeto. Implementamos la revocación buscando en la tabla global la entrada deseada y borrándola. Cuando se intenta un acceso, se encuentra que la capacidad apunta a una entrada ilegal de la tabla. Las entradas de la tabla pueden volver a utilizarse sin dificultad para otras capacidades, ya que tanto la capacidad como la entrada contienen el nombre único del objeto. El objeto para una capacidad y su entrada deben corresponder.

Claves: una clave es un patrón único de bits que puede asociase con cada capacidad, esta clave se define cuando se crea la capacidad, y no puede ser modificada ni inspeccionada por el proceso que posee dicha capacidad. Se puede definir una clave maestra con cada objeto o reemplazarse con una operación

Protección en un lenguaje

La especificación de la protección en un lenguaje de programación permite una descripción de alto nivel de políticas para la asignación y uso de recursos. Una implementación de lenguaje puede proporcionar el software para el cumplimiento de la protección cuando no está disponible una verificación automática soportada por el hardware.

Seguridad

El Sistema operativo es normalmente solo una porción del total de software que corre en un sistema particular. Pero el Sistema Operativo controla el acceso a los recursos del sistema. La seguridad de los Sistemas Operativos es solo una pequeña parte del problema total de la seguridad en los sistemas de computación, pero éste viene incrementándose en gran medida. Hay muchas razones para que la seguridad de los Sistemas Operativos reciba especial atención hoy en día.

La evolución de los sistemas de computación, ha sido en las últimas décadas de una magnitud asombrosa. Las computadoras se han tornado más accesibles, también se tiene un aumento en los riesgos vinculados con la seguridad. Pero hay una cosa que se ha mantenido constante a través de todo este tiempo, y es que los sistemas digitales se han vuelto cada vez más complejos. Los microprocesadores se han vuelto más complejos. Los sistemas operativos se han vuelto más complejos.

Los ordenadores se han vuelto más complejos. Las redes se han vuelto más complejas. Las redes individuales se han combinado y han aumentado todavía más su complejidad. Ejemplo claro de ello es Internet, la gran red de computadoras, a medida que aumenta su complejidad va tornándose más insegura.Si tenemos en cuenta que todo software no está libre fallos, entonces un software complejo es probable que falle y un porcentaje de estos fallos afecte a la seguridad.

También es importante mencionar que los sistemas complejos son necesariamente modulares, ya que de otra manera no se podría manejar su complejidad. Pero el aumento de la modularidad significa que la seguridad disminuye porque falla a menudo donde dos módulos se comunican.

La única manera razonable de probar la seguridad de un sistema es realizar evaluaciones de seguridad en él. Sin embargo, cuanto más complejo es el sistema, más dura se vuelve la evaluación de su seguridad. Un sistema más complejo tendrá más errores relacionados con la seguridad en su análisis, diseño y programación. Y desgraciadamente, el número de errores y la dificultad de evaluación no crece de acuerdo con la complejidad, crece mucho más rápido.

Cuanto más complejo es un sistema, más difícil es de entender. Hay toda clase de puntos de vulnerabilidad -interface entre usuario y máquina, interacciones del sistema- esto crece exponencialmente cuando no se puede mantener el sistema completo en la cabeza.

Cuanto más complejo es un sistema, más duro es hacer este tipo de análisis. Todo es más complicado: su análisis, su diseño, su programación y su uso.
Los sistemas operativos no escapan a esta realidad y se tornan cada vez más complejos. Un ejemplo es Microsoft Windows, que cuando se publicó en 1992 (Versión 3.1) tenía alrededor de 3 millones de líneas de código; Windows 95 alcanzó a los 15 millones y Windows 98 tiene 18 millones; Windows NT lanzado en 1992 tenía 4 millones de líneas de código; Windows NT 4.0 tiene 16.5 millones; Windows 2000 tiene entre 35 y 80 millones de líneas.
Como punto de comparación tenemos a Solaris que mantuvo su código fuente en aproximadamente 7 a 8 millones de líneas y Linux (Incluso con la suma del entorno gráfico X Windows y de Apache) que todavía se mantiene por debajo de los 5 millones de líneas.
En el pasado la seguridad física fue suficiente para resguardar un computadora contra ataques de intrusos, actualmente controles sofisticados deben instrumentarse para prevenir intentos de login desde terminales remotas y sobre otras redes de comunicación. Por último, cabe destacar que el

nivel de seguridad apropiado para un sistema en particular depende del valor de los recursos que se aseguran. Más información en

Seguridad Interna y Externa

La seguridad interna está relacionada a los controles incorporados al hardware y al Sistema Operativo para asegurar los recursos del sistema. La seguridad externa está compuesta por la seguridad física y la seguridad operacional. La seguridad física incluye la protección contra desastres (como inundaciones, incendios, etc.) y protección contra intrusos.

Seguridad Operacional

La seguridad operacional consiste en varias políticas y procedimientos implementados por el administrador del sistema de computación.
Mediante la autorización se determina qué acceso se permite y a qué entidad.
Como punto crítico se destaca la selección del personal y la asignación del mismo. Generalmente se dividen responsabilidades, de esta manera un operario no debe conocer la totalidad del sistema para cumplir con esas responsabilidades.
Se deben instrumentar diversos controles, y el personal debe saber de la existencia de dichos controles, pero desconocer cuáles son, para reducir la probabilidad de que intrusos puedan evadirlos.

Protección. Metas de la protección

Existen varios mecanismos que pueden usarse para asegurar los archivos, segmentos de memoria, CPU, y otros recursos administrados por el Sistema Operativo.
Por ejemplo, el direccionamiento de memoria asegura que unos procesos puedan ejecutarse solo dentro de sus propios espacios de dirección. El timer asegura que los procesos no obtengan el control de la CPU en forma indefinida. La protección se refiere a los mecanismos para controlar el acceso de programas, procesos, o usuarios a los recursos definidos por un sistema de computación. Seguridad es la serie de problemas relativos a asegurar la integridad del sistema y sus datos.

Hay importantes razones para proveer protección. La más obvia es la necesidad de prevenirse de violaciones intencionales de acceso por un usuario. Otras de importancia son, la necesidad de asegurar que cada componente de un programa, use solo los recursos del sistema de acuerdo con las políticas fijadas para el uso de esos recursos. Un recurso desprotegido no puede defenderse contra el uso no autorizado o de un usuario incompetente. Los sistemas orientados a la protección proveen maneras de distinguir entre uso autorizado y desautorizado.

Mecanismos y Políticas

El rol de la protección es proveer un mecanismo para el fortalecimiento de las políticas que gobiernan el uso de recursos. Tales políticas se pueden establecer de varias maneras, algunas en el diseño del sistema y otras son formuladas por el administrador del sistema. Otras pueden ser definidas por los usuarios individuales para proteger sus propios archivos y programas.

Las políticas son diversas, dependen de la aplicación y pueden estar sujetas a cambios a lo largo del tiempo.
Un principio importante es la separación de políticas de los mecanismos. ‘Los mecanismos determinan cómo algo se hará. Las políticas deciden que se hará’.
La separación es importante para la flexibilidad del sistema.

Vigilancia
La vigilancia se compone de la verificación y la auditoria del sistema, y la identificación de usuarios. En la vigilancia se utilizan sistemas muy sofisticados, a tal punto, que a veces pueden surgir problemas en la autentificación generando un rechazo al usuario legítimo.

Monitoreo de amenazas
Una manera de reducir los riesgos de seguridad es tener rutinas de control en el sistema operativo para permitir o no el acceso a un usuario. Estas rutinas interactúan con los programas de usuario y con los archivos del sistema. De esta manera, cuando un usuario desea realizar una operación con un archivo, las rutinas determinan si se niega o no el acceso y en caso de que el mismo fuera permitido devuelven los resultados del proceso.
Además las rutinas de control permiten detectar los intentos de penetración al sistema y advertir en consecuencia.

Amplificación
Como ya dijimos, los programas de vigilancia interactúan con los programas de usuario y los archivos del sistema. A veces estos programas (los primeros) requieren de más derechos de acceso de los que posee el usuario para realizar una operación determinada. Esto se conoce como amplificación.

Protección por contraseña
Existen tres clases principalmente de elementos que permiten establecer la identidad de un usuario:
Algo sobre las personas. Esto incluye huellas digitales, reconocimiento de voz, fotografía y firmas. Algo poseído por la persona. Esto incluye distintivos, tarjetas de identificación y llaves. Algo conocido por el usuario. Esto incluye contraseñas, nombre de la suegra, combinación de cerraduras. El esquema de autentificación más común es la simple protección por contraseña. El usuario elige una palabra que se le viene a la memoria, y la tipea de inmediato para ganar admisión al sistema de computación. Muchos sistemas no muestran la contraseña tal como ha sido ingresada (mostrar asteriscos en lugar de letras). La protección por contraseña es un esquema débil. En el sentido de que los usuarios tienden a elegir contraseñas fáciles de recordar. Entonces alguien que conoce al usuario podría intentar ingresar al sistema usando nombres de gente que la persona conoce. Esto puede resultar en una violación de la seguridad por los intentos repetitivos de ingreso. Algunos sistemas usan contraseñas cortas lo que facilita la conformación rápida de la lista de todas las posibles combinaciones. Los sistemas actuales utilizan contraseñas largas para frenar tales intentos de penetración.

Auditoria

La auditoria normalmente es realizada en sistemas manuales “después del hecho”. Los auditores son llamados periódicamente para examinar las transacciones recientes de una organización y para determinar si ha ocurrido actividad fraudulenta.
El registro de auditoria es un registro permanente de acontecimientos de importancia que ocurren en el sistema de computación. Se produce automáticamente cada vez que ocurren los eventos y es almacenado en un área protegida del sistema.
Las auditorias periódicas prestan atención regularmente a problemas de seguridad; las auditorias al azar ayudan a detectar intrusos.

Controles de acceso

Los derechos de acceso definen qué acceso tienen los sujetos sobre los objetos. Los objetos son entidades que contienen información, pueden ser físicos o abstractos. Los sujetos acceden a los objetos, y pueden ser usuarios, procesos, programas u otras entidades.
Los derechos de accesos más comunes son: acceso de lectura, acceso de escritura y acceso de ejecución. Estos derechos pueden implementarse usando una matriz de control de acceso.
Matriz de acceso

El modelo de protección del sistema se puede ver en forma abstracta como una matriz, la *matriz de acceso.*
Las filas de la matriz representan dominios (o sujetos) y las columnas representan objetos. Las entradas de la matriz consisten en una serie de derechos de acceso. Por ejemplo, la entrada $access_{(i,j)}$ define el conjunto de operaciones que un proceso, ejecutándose en el dominio D_i, puede invocar sobre un objeto O_j.

Objeto Dominio	F1	F2	F3	Lector de tarjeta	Impresora
D1	Read		Read		
D2				Read	Print
D3		Read	Execute		
D4	Read Write		Read write		

Políticas

El esquema de matriz de acceso provee el mecanismo para especificar una variedad de políticas. Se debe asegurar que un proceso que se ejecuta en el dominio D_i puede acceder sólo a aquellos objetos especificados en la fila i.
Las decisiones de política concernientes a la protección pueden implementarse por la matriz de acceso. Las decisiones políticas definen qué derechos deben estar incluidos en las entradas (i,j).
A veces decide el dominio de cada proceso ejecutable. Esta última política es generalmente decidida por el sistema operativo.
Los usuarios normalmente deciden el contenido de las entradas de la matriz de acceso. Cuando un usuario crea un nuevo objeto O_j, la columna O_j es agregada a la matriz de acceso con las entradas de inicialización apropiadas.

Criptografía

La criptografía es usada para la transformación de datos para hacerlos incomprensibles para todos, excepto para el usuario destinatario. El problema de la privacidad tiene relación con la prevención de la no autorización para la extracción de información desde un canal de comunicación. Los problemas de autentificación están relacionados con la prevención contra intrusos que intentan modificar una transmisión o insertar falsos datos dentro de una transmisión. Los problemas de disputa están relacionados con la providencia de reserva de un mensaje con prueba legal de la identidad enviada.

Sistema de privacidad criptográfico
En un sistema de privacidad criptográfico, el remitente desea transmitir cierto mensaje no cifrado a un receptor legítimo, la transmisión ocurre sobre un canal inseguro asume ser monitoreado o grabado en cinta por un intruso.

El remitente pasa el texto a una unidad de encriptación que transforma el texto a un texto cifrado o criptograma; el mismo no es entendible por el intruso. El mensaje es transmitido entonces, sobre un canal seguro. Al finalizar la recepción el texto cifrado pasa a una unidad de descripción que regenera el texto.

Criptoanálisis

Criptoanálisis es el proceso de intentar regenerar el mensaje desde el texto cifrado pero sin conocimiento de las claves de encriptación. Esta es la tarea normal de los intrusos. Si el intruso o criptoanalista no puede determinar un mensaje desde el texto cifrado (sin la clave), entonces el sistema de criptografiado es seguro.

Métodos y técnicas de encriptación

Está técnica consistía simplemente en sustituir una letra por la situada tres lugares más allá en el alfabeto esto es la A se transformaba en D, la B en E y así sucesivamente hasta que la Z se convertía en C.

Gronsfeld

Este método utiliza más de un alfabeto cifrado para poner en clave el mensaje y que se cambia de uno a otro según se pasa de una letra del texto en claro a otra.
Es decir que deben tenerse un conjunto de alfabetos cifrados y una forma de hacer corresponder cada letra del texto original con uno de ellos.

RSA

En los sistemas tradicionales de cifrado debe comunicarse una clave entre el emisor y el receptor del mensaje, el problema aquí es encontrar un canal seguro para transmitir dicha clave. Este problema viene a resolverse en los sistemas de clave pública la clave de cifrado, pues un tiempo enormemente de ordenador es necesario para encontrar una transformación de descifrado a partir de la de cifrado.

DES

DES fue desarrollado por IBM a mediados de los setenta. Aunque tiene un buen diseño, su tamaño de clave de 56 bits es demasiado pequeño para los patrones de hoy.
DES (Data Encryption Standard) es un mecanismo de encriptación de datos de uso generalizado. Hay muchas implementaciones de hardware y software de DES. Este transforma la información de texto llano en datos encriptados llamados texto cifrado mediante el uso de un algoritmo especial y valor semilla llamado clave. Si el receptor conoce la clave, podrá utilizarla para convertir el texto cifrado en los datos originales. Es un mecanismo de encriptado simétrico.

Chaffing & Winnowing

Esta técnica propuesta por Donald Rivest. Es más un intento de esquivar las restricciones a la criptografía en EE.UU. (y otros países) que una propuesta razonable debido al tamaño de los mensajes resultantes.

El término inglés “winnowing” se tomará como aventar es decir separar el grano de la paja y el término “chaffing” por el castellano empajar (cubrir o rellenar con paja). La idea básica consiste en mezclar la información real (grano) con otra de relleno (paja) de modo que sea imposible separarlas excepto para el destinatario.

SKIPJACK

Este algoritmo fue descalificado por el gobierno de Estados Unidos. Algunos detalles sobre el algoritmo en sí y sus aplicaciones en la práctica a los chips Clipper y Capstone. Skipjack fue desarrollado por la NSA inicialmente para los chips Clipper y Capstone. Su diseño comenzó en 1985 y se completó su evaluación en 1990.

BÍFIDO

El método Bífido es un cifrado fraccionario. Es decir que cada letra viene representada por una o más letras o símbolos, y donde se trabaja con estos símbolos más que con las letras mismas.

WLBYKYAAOTB

Este método altera la frecuencia de los caracteres a diferencia de lo que ocurre por ejemplo con los cifrados monoalfabéticos. Admite algunas variaciones como por ejemplo dividir la lista en 3,4,..., n partes.

Cifrado exponencial

Es un sistema basado en la exponenciación modular, debido Pohlig y Hellman (1978). Este método es resistente al criptoanálisis. Blowfish

Este algoritmo realiza un cifrado simple en 16 ciclos, con un tamaño de bloque de 64 bytes para un total de 448 bits. Aunque hay una fase compleja de la inicialización. El cifrado de datos es muy eficiente en los microprocesadores grandes.

Sistemas de clave pública

Un sistema criptográfico de clave pública es tan seguro como su clave. La distribución de las claves debe ser manejada sobre canales altamente seguros. Esto suele consumir mucho tiempo. A veces, tales canales de seguridad no están disponibles.
Los sistemas de clave pública no tienen tal problema en la distribución de la clave. En el sistema criptográfico convencional el cifrado y descifrado están íntimamente relacionados. Estos sistemas usan una clave encriptada, E, y una clave descifrado, D, de manera que no es computacionalmente viable (dentro de un tiempo razonable) determinar E a partir de D.

De esta forma, E puede ser hecha pública sin comprometer la seguridad de D. Esto simplifica el problema de la distribución de la clave. Cada usuario genera una clave de cifrado y una de descifrado, la clave de cifrado está hecha pública y la clave cifrada se mantiene secreta. Así cualquiera puede enviar un mensaje encriptado a un usuario particular (porque la clave de cifrado es pública), pero solo aquellos usuarios pueden descifrar el mensaje (porque la clave de descifrado es privada). E es llamada una clave pública y D es llamada una clave privada.

Firmas digitales

Para que una firma digital sea aceptada como sustituta de una firma escrita debe ser:

Fácil de autentificar (reconocer) por cualquiera.
Producible únicamente por su autor.

En los cripto-sistemas de clave pública el procedimiento es:
El remitente usa la clave privada para crear un mensaje firmado.

El receptor:
Usa la clave pública del remitente para descifrar el mensaje, guarda el mensaje firmado para usarlo en caso de disputas

Medidas básicas de seguridad

En general se puede afirmar que si la llave privada solo es conocida y accesible por el sujeto A, sería prácticamente imposible, para otro sujeto B, falsificar una firma digital del sujeto A, o abrir un sobre digital dirigido al sujeto A, utilizando métodos matemáticos. El atacante de un sistema va a

centrar su esfuerzo en encontrar debilidades en la implementación del software o hardware de seguridad. A continuación se mencionan los dos puntos de ataque más comunes:

Generación de números aleatorios

La generación de las llaves utiliza métodos pseudoaleatorios por lo que es muy importante que un sujeto B no puede replicar el procedimiento que siguió un sujeto A cuando este generó sus llaves.

Ataque a la Llave Privada

La llave privada, que normalmente reside en un archivo debe mantenerse encriptada con un algoritmo simétrico, utilizando como llave una contraseña. La contraseña debe ser elegida por el usuario en forma tal que resulte impredecible para quien intente adivinarlo por asociación de ideas. La encriptación por contraseña es normalmente presa de ataques denominados de diccionario que buscan exhaustivamente entre un conjunto de palabras formadas por letras del abecedario. Otro ataque más sutil se concentra en intentar por prueba y error las posibles contraseñas que un sujeto utiliza en base a acciones de ideas, por ejemplo, su apodo, el nombre de su esposa, su apodo y fecha de nacimiento, etc.

La llave privada solo se debe de encontrar desencriptada cuando está en la memoria de la computadora y mientras el programa de seguridad esté funcionando. Si el sujeto se encuentra en un entorno de cómputo en donde sea posible que un atacante realice un vaciado a disco del estado de la memoria del programa de seguridad, entonces la llave privada está en peligro. Si se está en un entorno de cómputo en donde sea posible que un atacante intercepte el teclado entonces, su llave privada está en peligro. Si se está en un entorno de cómputo en donde sea posible sustituir el programa de seguridad por uno falso que capture su contraseña y su llave privada encriptada entonces, su llave privada está en peligro.

Las formas más seguras de evitar el robo de llaves privadas es de firmar y abrir sobres en una computadora aislada física y virtualmente del mundo exterior. A dicha computadora deben de entrar mensajes a firmar y deben de salir mensajes firmados, nunca debe de salir ni exponer la llave privada. Este es el caso de por ejemplo, los Agentes Certificadores, Autoridades Certificadoras y en general aplicaciones altamente sensitivas.

BIBLIOGRAFÍA

1. **Silberschatz, Galvin, Gagne.**
   Sistemas Operativos
   México
   Limusa Wiley, 2002

2. **Flynn, Mchoes.**
   Sistemas Operativos
   México
   Thomson, 2001