UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE CONTADURÍA Y ADMINISTRACIÓN
DIVISIÓN SISTEMA UNIVERSIDAD ABIERTA Y EDUCACIÓN A DISTANCIA

APUNTE ELECTRÓNICO

Sistemas Operativos Multiusuarios

Licenciatura en Informática

SUAyED UNAM
Sistema Universidad Abierta y Educación a Distancia

Plan 2012 2016 actualizado

COLABORADORES

DIRECTOR DE LA FCA
Mtro. Tomás Humberto Rubio Pérez

SECRETARIO GENERAL
Dr. Armando Tomé González

– – – –

COORDINACIÓN GENERAL
Mtra. Gabriela Montero Montiel
Jefa del Centro de Educación a Distancia y Gestión del Conocimiento

COORDINACIÓN ACADÉMICA
Mtro. Francisco Hernández Mendoza
FCA-UNAM

– – – –

AUTOR
Lic. Salvador Meza Badillo

REVISIÓN PEDAGÓGICA
Mtro. Joel Guzmán Mosqueda

CORRECCIÓN DE ESTILO
Mtro. Carlos Rodolfo Rodríguez de Alba

DISEÑO DE PORTADAS
L.CG. Ricardo Alberto Báez Caballero
Mtra. Marlene Olga Ramírez Chavero

EDICIÓN
Mtra. Marlene Olga Ramírez Chavero

Dr. Enrique Luis Graue Wiechers
Rector

Dr. Leonardo Lomelí Vanegas
Secretario General

Mtro. Tomás Humberto Rubio Pérez
Director

Dr. Armando Tomé González
Secretario General

Mtra. Gabriela Montero Montiel
Jefa del Centro de Educación a Distancia
y Gestión del Conocimiento / FCA

Sistemas Operativos Multiusuarios
Apunte electrónico

Edición: 17 de octubre de 2014.

Facultad de Contaduría y Administración
Circuito Exterior s/n, Ciudad Universitaria
Delegación Coyoacán, C.P. 04510, México, Ciudad de México.

ISBN: 978-607-02-5790-2
Plan de estudios 2012, actualizado 2016.

Hecho en México

OBJETIVO GENERAL

El alumno conocerá los fundamentos de diseño y funcionamiento de un sistema operativo multiusuario, y será capaz de explotar sus servicios.

TEMARIO OFICIAL

(64 horas)

	Horas
1. Teoría de sistemas operativos	10
2. Windows Server	8
3. GNU/Linux	8
4. Free/BSD	8
5. Administración de archivos	8
6. Seguridad	8
7. Implantación de sistemas operativos	8
8. Tópicos avanzados de sistemas operativos	6
TOTAL	**64**

INTRODUCCIÓN

Sin el *software*, un equipo de cómputo no es más que un conjunto de dispositivos físicos sin ninguna utilidad; con el *software*, un equipo puede procesar, almacenar y manipular información así como realizar diversas actividades en beneficio de las personas e instituciones.

El *software*, para computadoras se clasifica, de manera general, en dos clases: los **programas de sistema,** que controlan la operación de la computadora, y los **programas de aplicación**, utilizados para tareas específicas y que ayudan a resolver los problemas de los usuarios.

El programa principal de una computadora es el sistema operativo, que administra todos los recursos de la computadora y proporciona la base sobre la cual pueden escribirse los programas de aplicación[1]. Existen diferentes categorías del sistema operativo: multitareas, monotareas, monousuario, por lotes, en tiempo real, tiempo compartido y multiusuarios: así mismo, existen sistemas operativos para diferentes plataformas, por ejemplo: Windows Server, GNU/Linux y Free/BSD, cada uno con sus propias características las cuales serán descritas a lo largo de la asignatura.

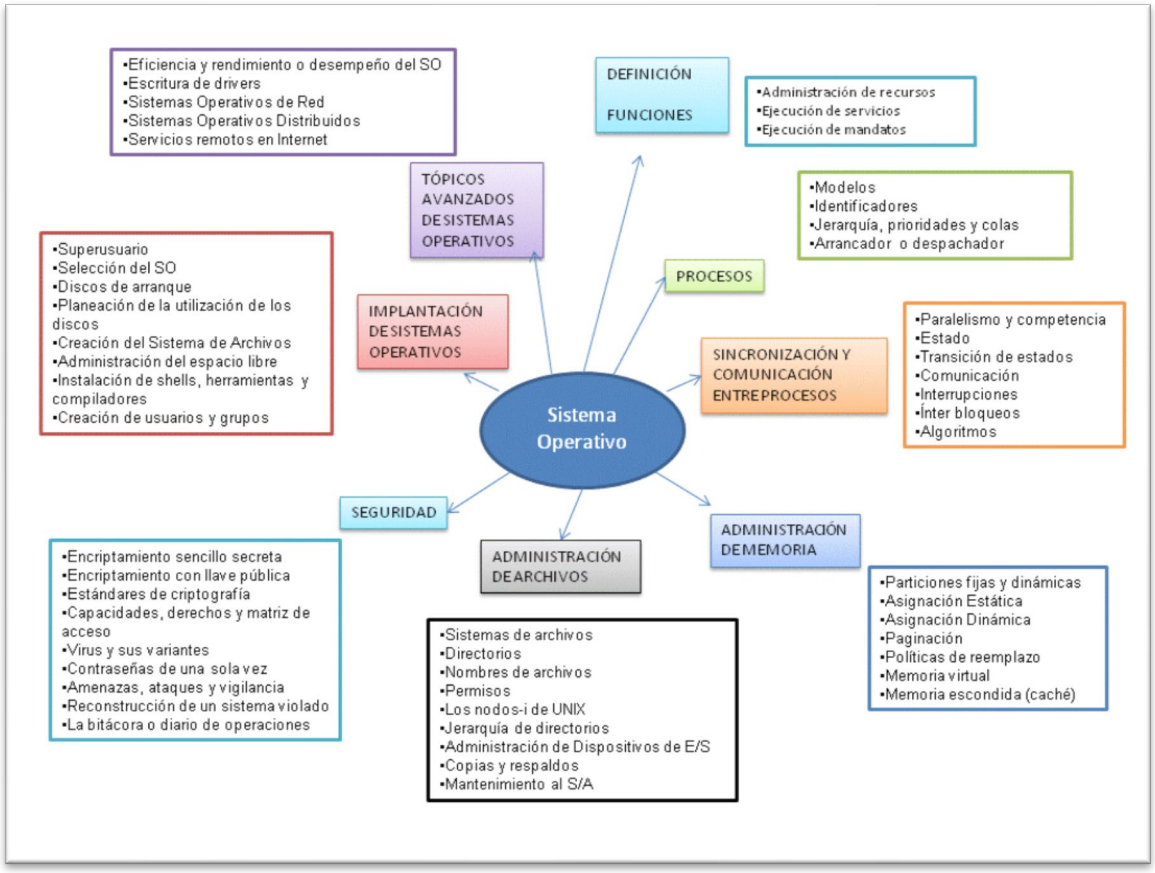
En los sistemas operativos modernos, la idea de multiusuario guarda el significado original de que puede utilizarse por varios usuarios a la vez, permitiendo la ejecución concurrente de los programas de aplicación. Las computadoras modernas utilizan múltiples procesadores y proveen las interfaces de usuario a

[1] Aunque también es importante tomar en cuenta que un sistema operativo no sólo está presente en las computadoras; sino en todo dispositivo electrónico que contenga un microprocesador, como celulares, *tablets*, videojuegos, etcétera, y su función es la misma.

través de una red de computadoras, e inclusive un grupo de computadoras pueden formar un *cluster* (agrupamiento de equipos), logrando altas capacidades de cómputo, como velocidad de procesamiento, tolerancia a fallas, escalabilidad, rendimiento, etcétera.

La materia de Sistemas operativos multiusuario se ha dividido en ocho unidades. En la primera se estudian los conceptos fundamentales aplicados a la teoría de los sistemas operativos multiusuario; en las unidades dos, tres y cuatro se estudian los aspectos más relevantes relacionados con los sistemas operativos: Windows Server, GNU/Linux y Free/BSD. La quinta unidad comprende la administración del sistema de archivos. En la sexta unidad se describen los conceptos y mecanismos que existen para la protección y seguridad de los sistemas operativos, y finalmente en la séptima y octava unidades, se describen las principales técnicas que se utilizan para la implantación de un sistema operativo.

ESTRUCTURA CONCEPTUAL



Licenciatura: Informática

UNIDAD 1

Teoría de sistemas operativos

SUAyED UNAM
Sistema Universidad Abierta y Educación a Distancia

Plan 2012 2016 actualizado

OBJETIVO PARTICULAR

El alumno identificará los conceptos más importantes que aplican a los sistemas operativos multiusuario, a su diseño y construcción, la importancia de los procesos cooperativos, así como el manejo y administración de la memoria.

TEMARIO DETALLADO

(10 horas)

1. Teoría de sistemas operativos

- 1.1. Definición de conceptos fundamentales
  - 1.1.1. Definición de sistema operativo multiusuario
  - 1.1.2. Funciones de los sistemas operativos multiusuario
- 1.2. Procesos
  - 1.2.1. Definición
  - 1.2.2. Modelos de procesos, e identificadores de procesos
  - 1.2.3. Jerarquía de procesos, prioridades y colas
  - 1.2.4. Arrancador o despachador de procesos
- 1.3. Sincronización y comunicación entre procesos
  - 1.3.1. Paralelismo y competencia entre procesos
  - 1.3.2. Estado de procesos
  - 1.3.3. Transición de estados

- 1.3.4. Comunicación entre procesos
- 1.3.5. Interrupciones
- 1.3.6. Interbloqueos de procesos
- 1.3.7. Algoritmos de administración de procesos
- 1.4 Administración de memoria
  - 1.4.1. Administración de la memoria
  - 1.4.2. Particiones fijas y dinámicas
  - 1.4.3. Asignación estática de la memoria
  - 1.4.4. Asignación dinámica de la memoria
  - 1.4.5. Paginación
  - 1.4.6. Políticas de reemplazo de páginas
  - 1.4.7. Memoria virtual
  - 1.4.8. Memoria escondida (*caché*)

INTRODUCCIÓN

El núcleo fundamental de una computadora es su sistema operativo; éste controla el *hardware*, carga las aplicaciones en la memoria, ejecuta esas aplicaciones y maneja los dispositivos y periféricos como discos e impresoras.

El objetivo principal del estudio de esta unidad, es conocer cómo funciona un sistema operativo, de tal modo, que pueda hacer que un sistema de cómputo sea utilizado de manera cómoda y eficiente por un usuario, además de familiarizarse con los procesos que intervienen en ello.

Los procesos y la comunicación entre el sistema operativo y el de cómputo son esenciales para la ejecución de programas. El sistema operativo mantiene, por cada proceso, una serie de estructuras de información para identificar sus características; así como los recursos que tiene asignados, tales como segmentos de memoria, puertos de comunicaciones, archivos abiertos, etcétera.

Además, para lograr el mayor aprovechamiento de las operaciones que se ejecutan en el sistema, es importante realizar una buena administración de la memoria, a través de la organización de los procesos y programas.

Así, pues, a lo largo de esta unidad, se abordarán temas que ayuden a lograr mejor comprensión del funcionamiento de un sistema operativo, conociendo las tareas que realiza, sus procesos y la forma en que éstos se comunican con el sistema; asimismo, se distinguirán los tipos de memoria que maneja, y la forma en que se administran sus tareas, archivos y recursos.

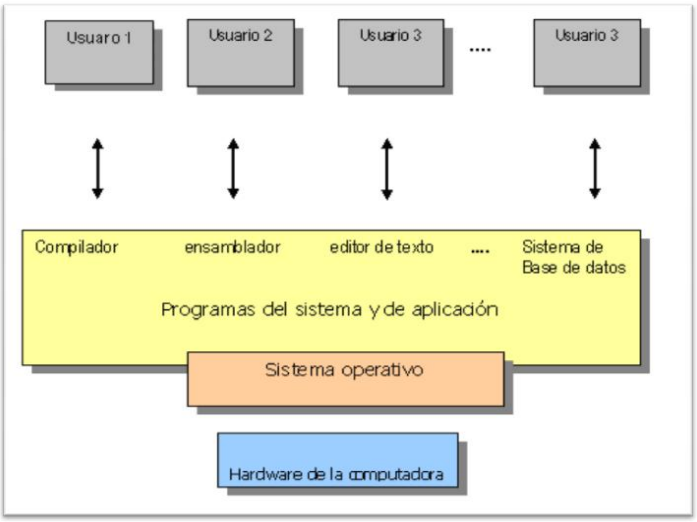
1.1. Definición de conceptos fundamentales

Como ya lo has visto en asignaturas previas, sabes que un sistema de cómputo está dividido en dos componentes principales: el *hardware* (dispositivos de E/S, CPU y memoria) y el *software* (sistema operativo y programas de aplicación).

Pero de acuerdo con el autor Silberschatz (2002: 4), cuando hablamos de un sistema de cómputo también estamos incluyendo, además de los dos mencionados, al usuario como un elemento esencial, ya que evidentemente por éste es por el que se realiza la comunicación entre el *software* y el *hardware*.

Para que este sistema de cómputo (PC) pueda funcionar para las tareas que se tienen previstas, emplea un conjunto de programas (*software*) que administran todos los recursos de la misma, permite la comunicación del usuario con el equipo y es el que sirve de plataforma para que se puedan instalar programas de aplicación (paquetería de Office, diseño, etcétera) conocido comúnmente como sistema operativo o *software* de sistema, el más importante en las PC.

En síntesis, un sistema de cómputo está compuesto por diferentes componentes como son: el *hardware*, el sistema operativo, el *software* de aplicación y de usuario y el elemento más importante que interactúa con los componentes, el usuario (Silberschatz, 2002: 4).



Vista abstracta de los componentes de un sistema de cómputo (Elaboración con base en Silberschatz, 2002: 4)

Equipo de cómputo
Máquina y equipo asociados con dispositivos de cómputo; tales como la unidad central de proceso (CPU), memoria, dispositivos periféricos, etcétera.
Programas de aplicación
Se denomina así al tipo de *software* que se utiliza para resolver los problemas de cómputo de los usuarios; ejemplo de ello son los procesadores de texto, hojas de cálculo, manejadores de bases de datos, navegadores de red, etcétera.
Usuarios
Son las personas autorizadas para utilizar un sistema de cómputo. Para acceder a los recursos del sistema, se autentifican a través de un nombre de usuario y una contraseña. Los usuarios pueden ser administradores del sistema y usuarios comunes que requieran administrar o utilizar los recursos de *hardware* y *software* disponibles y que cuenten con ciertos permisos y restricciones dentro del mismo.

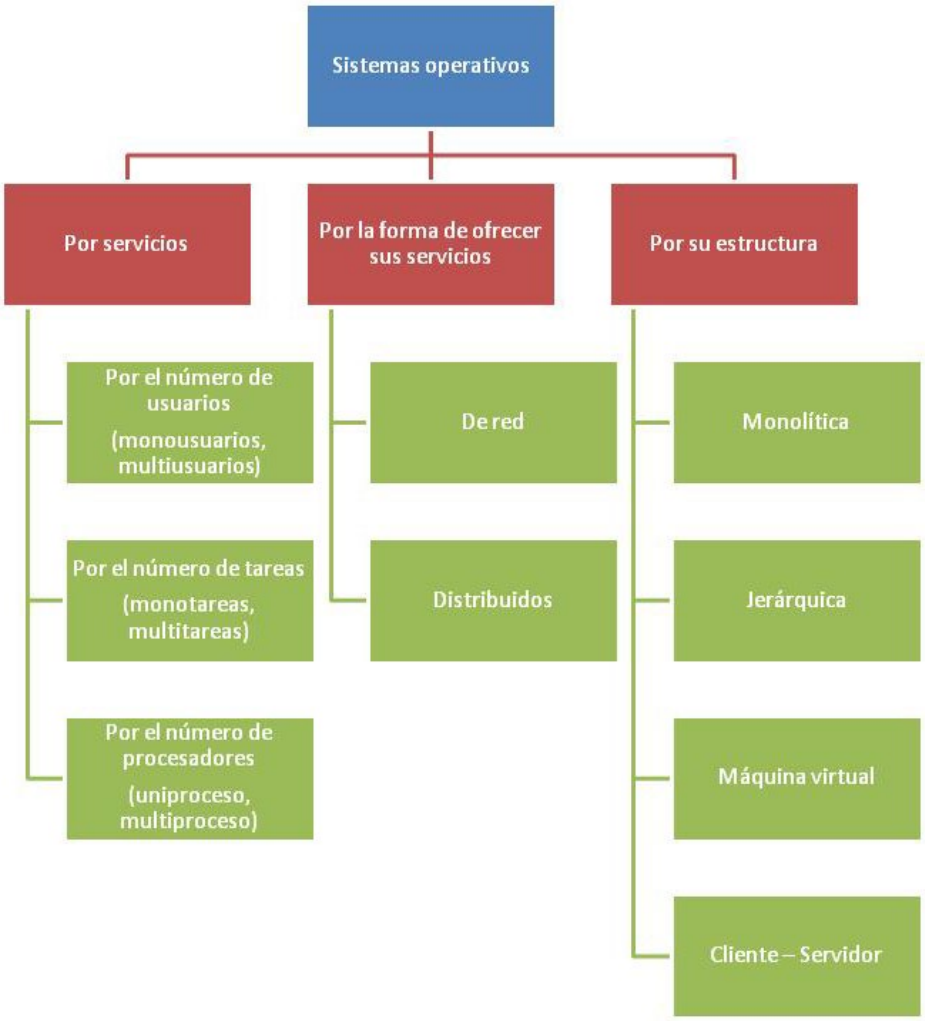
Sistema operativo
Conjunto de programas fundamentales que controlan y coordinan el *hardware* y los programas de aplicación de los usuarios. El sistema operativo proporciona los medios para el uso apropiado de los recursos en la operación del sistema de cómputo. Al igual que el gobierno, el sistema operativo por sí mismo no realiza alguna función útil, simplemente proporciona un ambiente dentro del cual otros programas pueden realizar un trabajo útil.

Entonces tenemos que un sistema de cómputo, necesita de un sistema operativo, que le ayuda a controlar y administrar los recursos de *hardware* y *software* (tiempo de procesamiento, espacio de memoria, espacio de almacenamiento para datos, dispositivos de entrada-salida, etcétera) sus procesos, así como la asignación de los recursos a usuarios y programas.

Debido a que pueden existir muchas solicitudes para usar los recursos de una computadora, el sistema operativo debe decidir a qué solicitudes les asignará recursos, espacio de memoria y prioridad, de manera que el sistema de cómputo pueda operar de manera eficiente y sin causar conflictos. Cada programa que se ejecuta en un sistema operativo, se conoce como *proceso* (que se profundizará en el siguiente tema).

En este rubro, encontramos dos tipos de sistemas operativos: monousuario, que soporta sólo las peticiones de un usuario limitándose a procesar una sola tarea a la vez (monotarea); y multiusuario, que es capaz de atender a más de un usuario al mismo tiempo, realizar varias tareas a la vez (multitareas) y compartir, incluso, los mismos recursos; de éste trataremos a lo largo de la materia, es el más común, debido a las diferentes actividades que se pueden realizar, incluso a través de las redes informáticas.

De forma más específica, los sistemas operativos se pueden clasificar como lo indica el siguiente esquema:



Esquema: Sistemas operativos. Elaboración propia.

1.1.1. Definición de sistema operativo multiusuario

Es más sencillo definir un sistema operativo por lo que hace, que por lo que es. El sistema operativo, como lo hemos visto, hace que las tareas de cómputo sean más sencillas para los usuarios.

Los sistemas operativos para computadoras de cualquier tamaño se ubican en diferentes tipos, que se distinguen por el tiempo de respuesta y la forma que se introducen los datos al sistema.

Pero los sistemas operativos, así como todo lo que se refiera al campo de la informática, va cambiando y evolucionando en el transcurso de los años. Es normal referirnos al sistema operativo tal y como lo conocemos hoy en día (multitarea, multiproceso, etcétera); pero el papel que tiene actualmente, no era el mismo de hace unas décadas. Para comprender mejor el funcionamiento del sistema operativo y la importancia de sus características actuales, echémosle una mirada a través del tiempo.

- **Breve historia de los sistemas operativos**

  → **Sistemas por lotes**

En la década de los 50, se implementaron las primeras computadoras que se operaban desde una consola. Los dispositivos de entrada comunes eran lectores de tarjetas y unidades de cinta, los dispositivos de salida eran impresoras de línea, unidades de cinta y perforadoras de tarjetas. El usuario no interactuaba directamente con este tipo de sistemas, sólo entregaba su trabajo al operador de la computadora, que los organizaba en lotes para su ejecución en el *mainframe*. Una característica importante de este tipo de sistemas es que comúnmente la unidad central de proceso (CPU) quedaba sin uso, debido a la velocidad de los dispositivos de entrada/salida (E/S), cuestión que se solucionó con el uso de discos de almacenamiento.

→ **Sistemas de tiempo compartido**

Los sistemas por lotes que utilizaron la multiprogramación proporcionaron en su momento un ambiente en el que los recursos del sistema (CPU, memoria,

dispositivos periféricos) se utilizaban eficazmente, con la limitante de que no ofrecían la interacción entre el usuario y el sistema de cómputo. En la década de los 80 surgen los sistemas de tiempo compartido o multitareas, que son una extensión lógica de la multiprogramación. La unidad central de proceso (CPU) ejecuta múltiples trabajos conmutando entre ellos, con la ventaja de que los usuarios pueden interactuar con cada programa mientras se encuentra en ejecución. Un sistema operativo de tiempo compartido planifica el uso de la unidad central de proceso (CPU) y la multiprogramación proporciona a cada usuario una pequeña parte de una computadora de tiempo compartido.

→ Sistemas para computadoras personales

Las computadoras personales (*Personal Computer* o PC) aparecieron a finales de la década de los años 70, y se referían a las microcomputadoras compatibles con las especificaciones de la empresa IBM. Este tipo de equipos fueron diseñados para ser utilizados por una persona a la vez, carecían de las características necesarias para proteger a un sistema operativo de los programas del usuario; los sistemas operativos para PC, por lo tanto, no eran ni multiusuario ni multitarea. Sin embargo, los objetivos de estos sistemas operativos han cambiado con el tiempo; en lugar de maximizar la utilización de la CPU y los dispositivos periféricos, los sistemas optan por maximizar la comodidad y grado de respuesta para el usuario, actualmente los sistemas operativos que utilizan los equipos PC son: Microsoft Windows, Macintosh de Apple, Linux, etcétera.

→ Sistemas paralelos

Entre los años de 1970-1980, aparecieron los sistemas paralelos, son aquellos que tienen más de un procesador (CPU) y están fuertemente acoplados compartiendo el *bus* (cables), el reloj y en ocasiones la memoria y los dispositivos periféricos, lo que les permite tener gran capacidad de realizar varias operaciones de manera

simultánea y manejar grandes volúmenes de información, del orden de los terabytes.

Sus características más importantes son:

- Mayor rendimiento.
- Mayor disputa por los recursos compartidos.
- Mayor trabajo en menor tiempo.
- Los procesadores pueden compartir periféricos, almacenamiento masivo y suministro de energía.
- Alta confiabilidad, la falla de un procesador no detendrá el sistema.

→ Sistemas de tiempo real

Desde la década de los 90 se han desarrollado los sistemas operativos de tiempo real, se caracterizan porque su parámetro clave es el tiempo y han sido perfeccionados para aplicaciones que requieren ser ejecutadas bajo ciertas restricciones de tiempo sobre la operación de un procesador o flujo de datos, que, en ocasiones, se emplea como dispositivo de control en aplicaciones delicadas. Por ejemplo, en los sistemas industriales, en el ámbito de la investigación científica, en el campo de la medicina, etcétera.

Una de las características de un sistema de tiempo real es que tiene definidas y fijas ciertas restricciones de tiempo. El procesamiento debe realizarse dentro de los límites de ese tiempo, de lo contrario, el sistema fallará.[2]

[2] Meza Badillo, S. (n.d.) *Sistemas operativos multiusuarios*. Fecha de recuperación 05 de diciembre de 2012, en: http://fcasua.contad.unam.mx/apuntes/interiores/docs/98/2/sis_operativos.pdf p.13.

→ Sistemas distribuidos

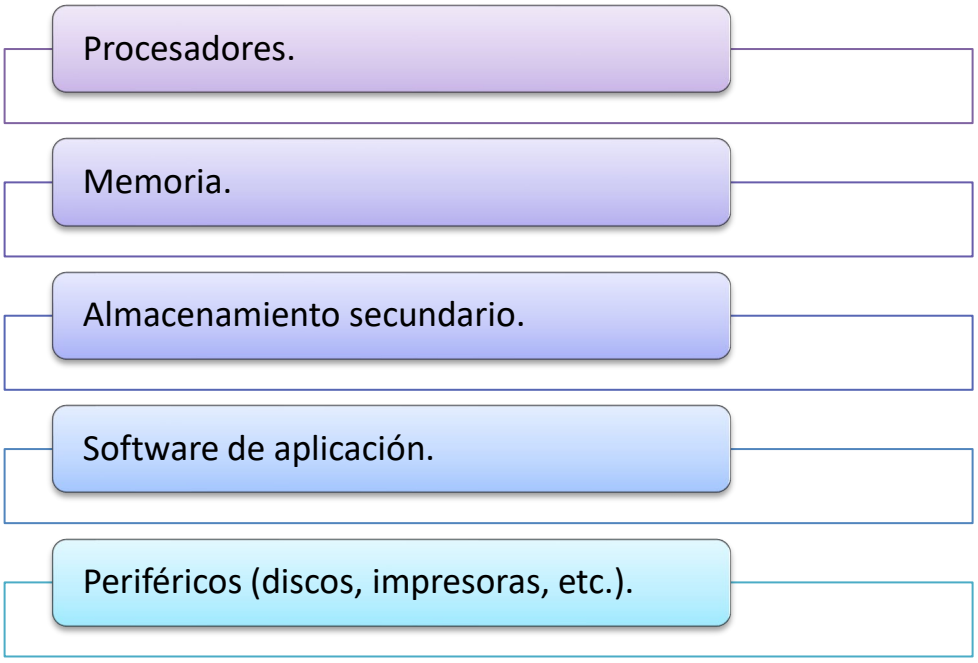
En la década de los 70, y debido al desarrollo tecnológico de los procesadores, el crecimiento de las redes de área local (LAN) y de las telecomunicaciones, permitieron conectar computadoras para la transferencia de datos a alta velocidad. Esto dio origen al concepto de "*Sistemas distribuidos*", que tiene como ámbito el estudio de redes, como, por ejemplo: Internet, redes corporativas, redes de telefonía móvil, etcétera. Hoy, todas las computadoras personales (PC), estaciones de trabajo y dispositivos móviles avanzados cuentan con la capacidad para ejecutar aplicaciones en red. Los sistemas operativos actuales como Windows, Linux, Android, etcétera incluyen la *suite* de protocolos TCP/IP y otros, que permiten a los equipos y dispositivos móviles tener acceso a Internet mediante diversas tecnologías de acceso (Redes LAN, WiFi, 3G, etcétera).

Sus características más importantes son:

Concurrencia (los recursos en la red pueden ser usados simultáneamente).
Carencia de reloj global (la realización de una tarea es distribuida a los componentes).
Fallas independientes (si un componente falla, los demás siguen funcionando).

De acuerdo con lo visto, los sistemas operativos multiusuario han evolucionado tomando en cuenta varios conceptos de la amplia variedad de los mismos, descritos en la presente unidad, desde las grandes computadoras centralizadas, hasta las computadoras personales de procesamiento distribuido y conectadas en red, los sistemas multiusuario son el resultado de esta evolución. Los sistemas operativos multiusuario dan la posibilidad de que dos o más usuarios compartan los mismos recursos de forma simultánea de manera local o remota a través de las redes de comunicación.

La compartición de recursos es principalmente:



Lo que reduce el tiempo ocioso del uso de los procesadores, costos de energía y equipamiento para los usuarios. Los sistemas operativos modernos, soportan funciones multiusuario. Por ejemplo: Unix, Solaris, Linux, FreeBSD, Windows Server en sus diferentes versiones.

1.1.2. Funciones de los sistemas operativos multiusuario

Las funciones básicas del sistema operativo se agrupan en tres rubros:

I. Gestión de los recursos de la computadora

En una computadora existen varios programas, en donde puede haber uno o más usuarios, con la ejecución simultánea de los programas, que compiten por los recursos del equipo, y es el sistema operativo el que administra su asignación y uso, éste debe garantizar la protección de los programas y suministrar información sobre el uso de los recursos que asigna.

II. Ejecución de servicios para los programas

El sistema operativo da a los programas un conjunto de *servicios*, o *llamadas*, que pueden solicitarse cuando lo necesiten, proporcionando a los programas una visión de máquina extendida.

III. Ejecución de comandos de los usuarios.

El módulo del sistema operativo que permite que los usuarios dialoguen de forma interactiva con el sistema operativo, a través de una *interfaz* (línea de comandos o gráfica), es el intérprete de comandos conocido como *Shell*.

1.2. Procesos

Los primeros sistemas de cómputo permitían la ejecución de un programa a la vez, las computadoras modernas de hoy día pueden ejecutar varios programas al mismo tiempo, por ejemplo: mientras ejecutan un programa, también pueden leer un disco, leer un dispositivo externo, abrir un navegador, mandar un archivo a la impresora, etcétera. Para lograr esto, se requirió de mayor control y una mayor división de los diferentes programas y dio por resultado el concepto de *proceso*. A continuación, se presentan los conceptos más importantes aplicados al proceso en un sistema de cómputo.

1.2.1. Definición

Un proceso se define como un programa de ejecución, y es la unidad de procesamiento gestionada por el sistema operativo. Como dice Carretero:

> "El sistema operativo mantiene, por cada proceso, una serie de estructuras de información para identificar las características de éste, así como los recursos que tiene asignados. (…). El sistema operativo mantiene una tabla de procesos con todos los bloques de control de proceso (BCP). Por razones de eficiencia, la tabla de procesos se construye normalmente como una estructura estática, que tiene un determinado número de BCP" (Carretero, 2001: 78).

El proceso no incluye información de entrada-salida (E/S), ya que esto está reservado al sistema operativo.

1.2.2. Modelos de procesos e identificadores de procesos

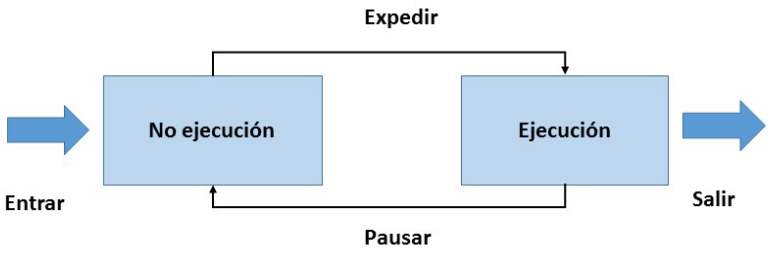
Se entiende por modelos de procesos, una representación de los estados que puede tener un proceso, considerando los siguientes: nuevo, ejecución, en espera, listo y terminado.

En estos modelos (Tanenbaum, 2003: 72-73), todo el *software* que se ejecuta en una computadora está organizado en procesos secuenciales; y cada uno de ellos posee una unidad central de proceso de forma virtual, que también es conocida como paralelismo virtual; es decir, el verdadero CPU cambia en forma continua de un proceso a otro, a esta conmutación se le llama multiprogramación, permitiendo aprovechar al máximo la memoria y finalizar con más tareas en menos tiempo. Cuando un proceso se está llevando a cabo y se bloquea esperando por la E/S, en la CPU se envía una instrucción para que se ejecute otro proceso (no de forma paralela; sino intercalada), ayudando a hacer más eficiente el procesamiento de programas, a esto se le llama *concurrencia*.

Entre los modelos de procesos tenemos, el de dos y el de cinco estados.

→ Modelo de dos estados.

Es el modelo más simple. Consiste en que un proceso puede estar ejecutándose o no; cuando se crea un nuevo proceso, se pone en estado de `no ejecución`. En algún momento, el proceso que se está ejecutando pasará al estado `no ejecución` y otro proceso se elegirá de la lista de procesos listos para ejecutar y ponerlo en estado `ejecución`. Como se muestra en la siguiente ilustración:



Elaboración con base en: http://www.sitioproyectos.com/ms-dos/Conceptos%20basicos%20sobre%20procesos.htm (27/04/2016)

→ Modelo de cinco estados.

En este modelo se necesita un estado en donde los procesos permanezcan esperando la realización de la operación de Entrada/Salida por parte del Sistema Operativo hasta que puedan proseguir. Se divide entonces al estado `no ejecución` en dos estados: `listo y espera`, y se agrega, además, un estado de `nuevo` y otro `terminado`.

Los cinco estados se muestran en el siguiente diagrama:

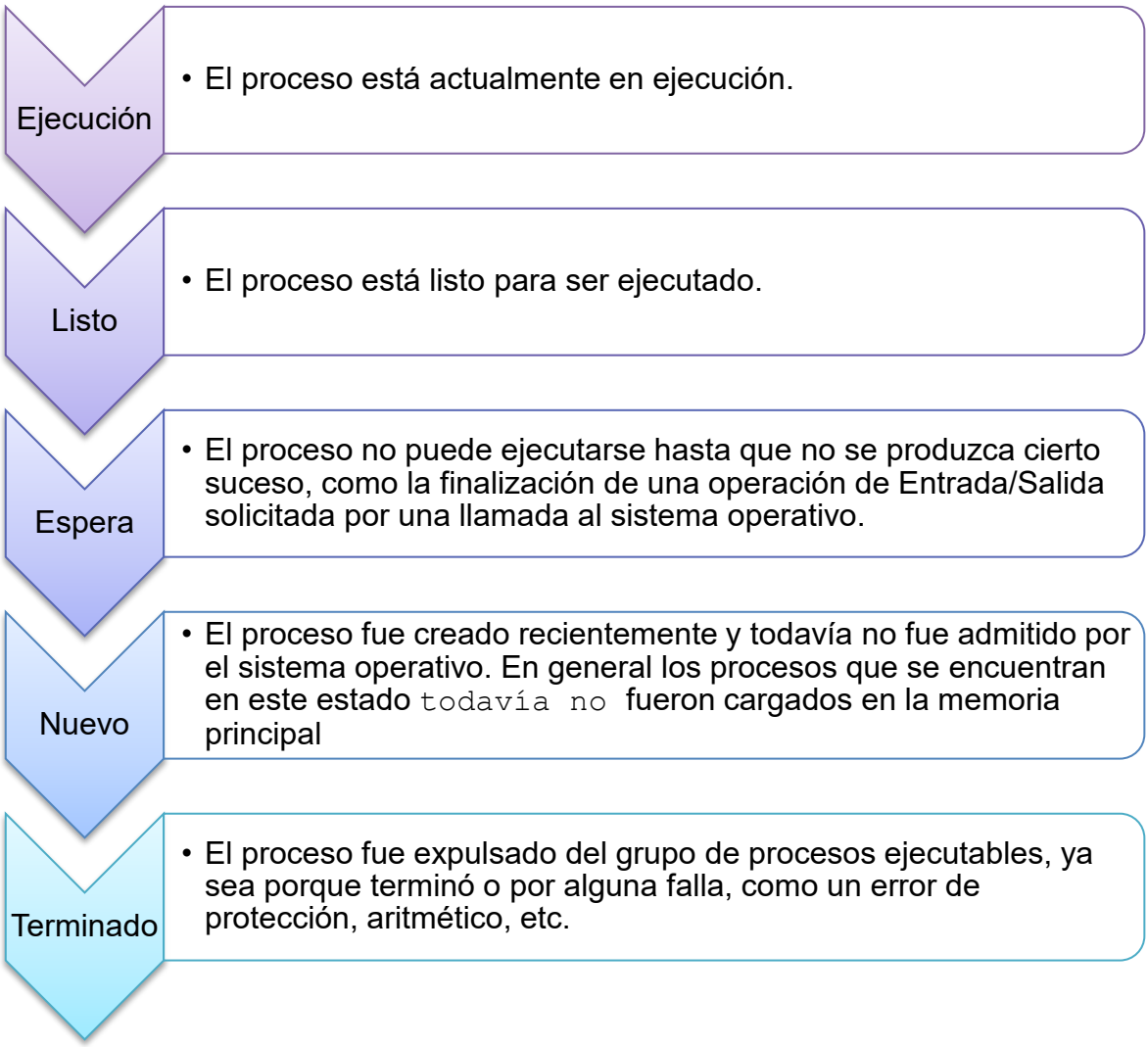


Diagrama de cinco estados. Elaboración propia.

Un ejemplo de diagrama de los cinco estados lo puedes consultar en:

http://www.sitioproyectos.com/ms-dos/Conceptos%20basicos%20sobre%20procesos.htm

En el punto 1.1.3.

Estados suspendidos.

Además de los estados básicos de los procesos (ejecución, listo y bloqueado), éstos pueden estar en los estados de *espera* y *suspendido*. En este modelo, dos o más procesos suspendidos (*HOLD*) pueden cooperar mediante señales de forma que uno obliga a detenerse a los otros, hasta que reciban una señal para continuar. Es frecuente que existan procesos en espera (*batch*) para que puedan ser ejecutados cuando así lo decida el sistema operativo; es decir, el sistema operativo analiza este tipo de procesos para ejecutarlos cuando disponga de los recursos necesarios. Al suspender los procesos el sistema operativo, les retira los marcos de página y los deja en una zona de intercambio. El objetivo de que suspenda los procesos, es que debe dejar disponible la memoria suficiente a los procesos que no están suspendidos para evitar la hiperpaginación (alta paginación), que ocasiona problemas de rendimiento. Es importante mencionar que no todos los sistemas operativos manejan la opción de `suspensión`; por ejemplo, los sistemas monousuario pueden no manejar la suspensión, por lo que el usuario del sistema deberá terminar los procesos, si observa que no se ejecutan de manera correcta.

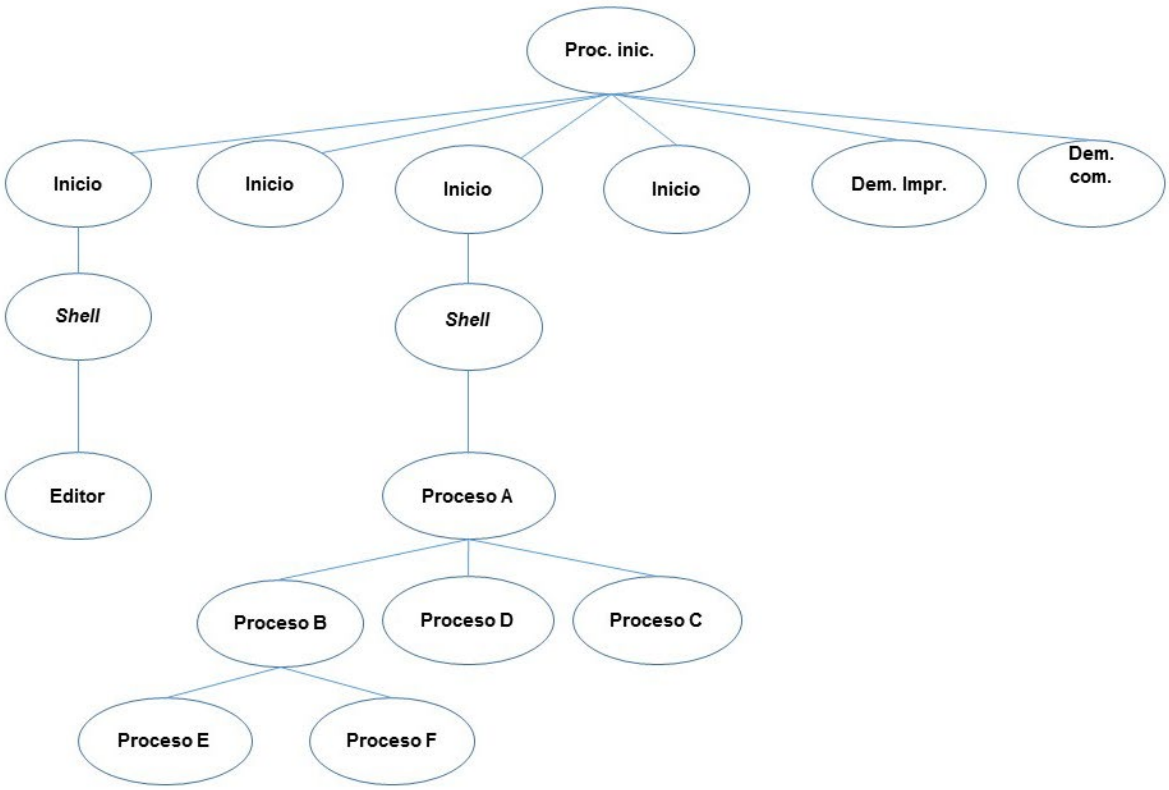
A continuación, se presenta un ejemplo de cómo es su funcionamiento:

- ✓ Por lo general, se utiliza una variable de tipo “semáforo”, por la que se da un intercambio de señales y se sincronizan todos los procesos.
- ✓ Si el proceso está en espera de una señal, se suspende (*Hold*) hasta que la señal se envíe (*SIGNAL)*.
- ✓ Se mantiene una lista (*batch)* de procesos en espera en el semáforo.
- ✓ El sistema operativo, elige los procesos de la lista que está en espera (*batch*) por medio del uso de la política *FIFO* (*First In First Out).*

Identificadores de procesos.

Los procesos se identifican mediante su identificador de proceso. Un proceso nuevo se crea por la llamada al sistema *fork* (*bifurcar*) y puede tener procesos hijos; el

proceso creador se denomina *proceso padre* y a los nuevos se les denominan *procesos hijos*. Los procesos nuevos pueden crear otros y formar un árbol de procesos. Para cada proceso existe un padre, pero puede tener varios hijos, como se muestra en el siguiente esquema:



Jerarquía de procesos. Elaboración con base en: (Carretero, 2001: 79)

De acuerdo a lo que plantea Durán, un proceso nuevo se puede ejecutar de la siguiente forma:

1. El proceso padre continúa ejecutándose de manera concurrente con sus procesos hijo.
2. El proceso padre espera hasta que alguno o todos sus procesos hijo hayan concluido.

Hay dos posibilidades más en términos del espacio de direcciones del nuevo proceso:

1. El proceso hijo es una copia del proceso padre.
2. Se carga un programa en el proceso hijo. (Durán, 2007: 7)

Por ejemplo, en el sistema operativo UNIX, el proceso nuevo consiste en una copia del espacio de direcciones del proceso original; este mecanismo permite que el proceso padre se comunique fácilmente con su proceso hijo. Ambos procesos (el padre y el hijo) continúan su ejecución en la instrucción que va después de la llamada *fork*, con una diferencia: el código de retorno para la llamada *fork* es cero para el proceso nuevo (hijo), en tanto que el identificador de proceso (distinto de cero) del hijo se devuelve al padre (Silberschatz, 2002: 97-98).

1.2.3. Jerarquía de procesos, prioridades y colas

> "Un proceso se representa por un conjunto de datos denominado *bloque de control de procesos* (PCB), los que permiten al sistema operativo localizar información sobre el proceso y mantenerlo registrado por si hay que suspender la ejecución temporalmente."[3]

Información del bloque de control de procesos:

- Información de identificación
- Información del estado de la CPU
- Información del control del proceso
- Información de uso de recursos

[3] Meza Badillo, S. (n.d.) *Sistemas operativos multiusuarios*. Fecha de recuperación 05 de diciembre de 2012, en: http://fcasua.contad.unam.mx/apuntes/interiores/docs/98/2/sis_operativos.pdf p. 22

Para profundizar en los puntos anteriores, se te sugiere consultes la página siguiente:

http://www.atc.uniovi.es/telematica/2ac/Apuntes-y-Ejercicios/T08-Procesos.pdf. En ella encontrarás mayor información.

La información de control del proceso incluye los siguientes puntos:

- Estado de proceso.
- Estadísticas de tiempo y uso de recursos.
- Ocupación de memoria interna y externa (*swapping*).
- Recursos en uso.
- Archivos en uso.
- Privilegios.

Los bloques de control de proceso se almacenan en colas y hay una por cada estado posible de los procesos, éstos se dividen como se muestra a continuación:

Activos	Inactivos
Son aquéllos que compiten por el procesador o están en condiciones de hacerlo: ◆ Ejecución. Cuando un proceso tiene el control del procesador.	Son aquéllos que no pueden competir por el procesador; pero pueden volver a hacerlo si se soluciona el problema que los ha dejado en “suspenso” (falla de un dispositivo de entrada-salida (E/S):

◆ Preparado. Son aquellos procesos que están dispuestos a ser ejecutados. ◆ Bloqueado. No pueden ejecutarse porque requieren algún recurso no disponible o están en condiciones de hacerlo.	◆ Suspendido bloqueado. Proceso que ha sido suspendido y que, además, está a la espera de un evento para desbloquearse. ◆ Suspendido preparado. Proceso que ha sido simplemente suspendido. (Panizo, 2002: 1-12)

→ Prioridades

A cada proceso se le asigna una prioridad en función de la urgencia y de los recursos que disponga, lo cual determina la frecuencia de acceso al procesador. Esto ayuda a los usuarios a aprovechar los recursos del sistema de manera eficiente y minimizar los tiempos de respuesta del equipo.

De acuerdo con Pérez Campanero (Pérez, 2002: 71), los tipos de prioridades que se pueden manejar son:

- *Asignadas por el sistema operativo*

  No interviene directamente el usuario; sino que el mismo SO establece la prioridad al comenzar la ejecución del proceso, pudiendo asignarlas racionalmente (reconociendo los privilegios del propietario y del modo de ejecución), o bien arbitrariamente (conforme van llegando los procesos).

- *Asignadas por el propietario*

  Son determinadas por el usuario antes de comenzar la ejecución, pero de no organizarse adecuadamente pueden provocar algunos errores dentro del sistema, y hasta podría perderse el trabajo que se haya realizado.

Aunado a estas dos prioridades, establecidas al inicio de la ejecución del proceso, también están otras dos que se hacen cuando el proceso ya se está desarrollando:

- *Estática*

  Se caracteriza porque no puede ser modificada durante la ejecución.

- *Dinámica*

  Esta puede ser modificada en función de los eventos que se produzcan.

Los procesos, en los diferentes estados que tienen, son agrupados en *listas* o *colas* **(***lista de procesos del sistema -job queue):*

En esta lista están todos los procesos del sistema, al crearse un proceso nuevo se agrega a la misma el PCB; cuando el proceso termina su ejecución, es borrado.

Existen dos tipos de colas:

- *Cola de procesos listos (ready queue)*

  Esta cola se compondrá de los procesos que estén en estado `listo`; la estructura de la cola dependerá de la estrategia de planificación utilizada.

- *Cola de espera de dispositivos (device queue)*

  Los procesos que esperan por un dispositivo de E/S en particular son agrupados en una lista específica al dispositivo. Cada dispositivo de E/S tendrá su cola de espera.

1.2.4. Arrancador o despachador de procesos

El componente implicado en la función de la planificación de la unidad central de proceso (CPU) es el *despachador* (Silbertschatz, 2002: 139-141). Este componente inicia la ejecución de los procesos, es el módulo que da el control del CPU al proceso seleccionado por el planificador de corto plazo[4] y comprende las siguientes funciones:

- Conmutación de contexto (el CPU conserva el estado de un proceso anterior y carga el estado guardado de otro proceso).
- Conmutación a modo de usuario.
- Apuntar a la localidad apropiada en el programa del usuario para reinciar el programa.

> "El despachador es uno de los módulos del administrador de procesos y decide a qué procesador asignar el proceso que tiene que ser ejecutado; éste deberá ser muy rápido, ya que es invocado en cada conmutación de procesos. El tiempo que le lleva al despachador detener un proceso e iniciar la ejecución de otros, se conoce como *latencia de despacho*." (Meza, n.d: 24).

[4] Ver *Tipos de planificador* (Glosario).

1.3. Sincronización y comunicación entre procesos

1.3.1. Paralelismo y competencia entre procesos

Como vimos en el tema 1, en los sistemas paralelos existen varios procesadores que permiten realizar distintas operaciones de manera simultánea y manejar grandes volúmenes de información.

Un programa concurrente es visto como una colección de procesos secuenciales autónomos que se ejecutan (lógicamente) en paralelo. La concurrencia puede darse en sistemas con un solo procesador o con varios procesadores que compartan algo en común. La ejecución de procesos toma una de las siguientes formas:

"--Multiprogramación: ejecución de múltiples procesos en un solo procesador.
-- Multiprocesamiento: ejecución de múltiples procesos en un sistema multiprocesador donde hay acceso a la memoria compartida.
-- Programación distribuida: ejecución de múltiples procesos en varios procesadores, los cuales no comparten memoria." [5]

Cuando dos o más procesos intentan emplear un mismo recurso al mismo tiempo, se da una condición de competencia entre ellos, pudiendo ser de la siguiente forma:

-- Compatibles: Son utilizados por varios procesos de forma concurrente.

[5] Pedro Mejía Álvarez: *Procesos concurrentes*, material electrónico disponible en: http://delta.cs.cinvestav.mx/~pmejia/capi5tr.ppt, diapositiva 6/19. Recuperado el 21/04/16.

-- No compatibles: su uso se restringe únicamente a un solo proceso.

Evidentemente, en esta competencia, un proceso se ejecutará y el otro deberá esperar, incluso puede llegar a no ser concluido exitosamente.

1.3.2. Estado de procesos

Cuando se ejecuta un proceso, cambia su estado. El estado de un proceso se define principalmente por la actividad que realiza el proceso. Cada proceso puede tener uno de los siguientes estados, de acuerdo a como lo plantea Silbertschatz (2002: 88-89):

* **Nuevo**: el proceso se está generando.
* **Ejecución**: se están ejecutando instrucciones.
* **En espera**: proceso que espera a que ocurra algún evento (por ejemplo, la terminación de una operación de E/S).
* **Listo**: el proceso está en espera de ser asignado a un procesador.
* **Terminado**: el proceso ha terminado su ejecución. [6]

1.3.3. Transición de estados

De acuerdo con Carretero (2001: 97), la activación del sistema operativo se realiza mediante las “interrupciones”. Cuando se produce una interrupción se llevan a cabo las siguientes operaciones:

- Se guarda el estado del CPU en el correspondiente BCP.
- Se procede a ejecutar la rutina de tratamiento de interrupción del sistema operativo.

[6] Pedro Mejía Álvarez: *Procesos concurrentes*, material electrónico disponible en:http://delta.cs.cinvestav.mx/~pmejia/capi5tr.ppt, diapositiva 8/19. Recuperado el 21/04/2016.

Como resultado de lo anterior, se pueden producir cambios en el estado de algunos procesos, aunque no necesariamente. Por ejemplo, si un proceso **A** está bloqueado y espera a que termine una lectura de disco y llega una interrupción del mismo, entonces se produce un cambio y entra en acción el sistema operativo para manejar la interrupción. Si se indica que la interrupción terminó por lo que esperaba **A**, el sistema operativo cambiará el estado de este proceso a `listo` o a `ejecución` si así lo decide el módulo de planificación de procesos.

El sistema operativo requiere realizar un trabajo muy cuidadoso para que se ejecuten de manera correcta las transiciones de los estados de los procesos. Uno de los puntos más delicados tiene que ver con los registros que efectúa un equipo y son:

- Si un proceso está en ejecución, parte de su información se encuentra en los registros del equipo, éstos se modifican constantemente por la ejecución de instrucciones de máquina.
- El proceso detiene su ejecución porque llega una interrupción o porque el proceso solicita un servicio del sistema operativo.
- El sistema operativo ejecuta de manera inmediata una interrupción o un servicio demandado.
- La ejecución del sistema operativo, modifica los contenidos de los registros del equipo y destruye sus valores anteriores.

Los siguientes son algunos tipos de transiciones, como lo propone Juan Pérez Campanero:

a) **Comienzo de la ejecución**. Cuando el usuario da la orden para la ejecución del programa, el sistema operativo crea un proceso.

b) **Paso a estado de ejecución**. Cuando el procesador está libre y en la cola de preparados haya algún proceso, se pondrá en ejecución el primero de ellos.

c) **Paso a estado bloqueado**. Un proceso que se encuentre en ejecución puede solicitar al sistema quedarse en espera por algún otro evento, por ejemplo, la terminación de una operación de entrada/salida.

d) **Paso a estado preparado**. Que puede estar por cuatro causas:

- "Ejecución por el usuario de un programa. (…)

Producción del evento, por el que estaba bloqueado el proceso. Si el proceso estaba bloqueado pasará de la cola de procesos bloqueados a la de preparados. De esta manera, vuelve a competir por el uso del procesador. Si no existiera ninguno en el procesador, el proceso desbloqueado pasará automáticamente a ejecución.".

"- *Interrupción*. Si se produce una interrupción que fuerza al sistema operativo a ejecutar otro proceso, el que está en ejecución pasará a la cola de preparados y puede seguir compitiendo por el uso del procesador. (…)

- *Reanudación* de un proceso suspendido preparado.

e) Paso a estado suspendido preparado. Este paso también puede darse por varias causas:

- Ejecución por el usuario. El usuario ejecuta una orden al sistema operativo para suspender la ejecución de un programa.

- Producción del evento por el que estaba bloqueado el proceso. Si el proceso estaba suspendido bloqueado y se produce el evento, pasará a suspendido preparado.

- Ejecución de una llamada al sistema operativo, por lo que el proceso se suspende a sí mismo.

f) Paso a estado suspendido bloqueado. Este paso se da porque el usuario dé una orden al sistema operativo para que suspenda la ejecución del programa y se diera por casualidad de que el proceso correspondiente a dicho programa estaba en estado bloqueado. El proceso permanecerá en este estado hasta que se reanude por una orden del usuario o del sistema operativo, o bien se produzca el evento por el que está bloqueado, pasando en este caso ha Suspendido Preparado". (Pérez, 2002: 85-86).

1.3.4. Comunicación entre procesos

Los procesos concurrentes que se ejecutan en el sistema operativo pueden ser de dos tipos: *independientes* y *cooperativos*. Los primeros no pueden afectar ni verse afectados por otros procesos que se estén ejecutando, los segundos sí pueden afectar o verse afectados por otros que se estén ejecutando en el sistema; sin embargo, existen procesos que comparten datos con otros y se denominan *procesos cooperativos*. Éstos pueden comunicarse en un ambiente de memoria compartida; es decir, el sistema permite crear segmentos de memoria a los que pueden acceder múltiples procesos.

Un servicio de comunicación entre procesos (IPC) proporciona un mecanismo para que éstos se comuniquen y para sincronizar sus acciones, sin que compartan un mismo espacio de direcciones. El IPC es muy útil en un ambiente distribuido, en donde los procesos que se comunican pueden residir en diferentes equipos

conectados en red. El programa “chat” que se utiliza en internet, es un ejemplo de este tipo de comunicación entre procesos.

El servicio IPC realiza al menos dos operaciones: envío y recepción de mensajes (*send, receive*). Por ejemplo, si dos procesos, A y B, requieren comunicarse entre sí, debe existir un enlace de comunicación de tipo lógico. A continuación, se describen algunos métodos para implementar de manera lógica, un enlace, utilizando las operaciones *send-receive*.

Para lograr la comunicación entre los procesos, se requieren referencias entre sí, por lo cual se utiliza una comunicación directa o indirecta.

→ Comunicación directa

En este tipo de comunicación cada proceso que quiere comunicarse, debe nombrar de manera explícita al receptor o al emisor de la comunicación.

Ejemplo:

- *send* (**A**, mensaje). Enviar un mensaje al proceso A.
- *receive* (**B**, mensaje). Recibir un mensaje del proceso B.

En la comunicación, se tienen las siguientes propiedades:

- Un enlace se establece de manera automática entre cada par de procesos.
- Un enlace se asocia de manera exacta con dos procesos.
- Entre cada par de procesos existe de manera exacta sólo un enlace.

→ Comunicación indirecta

En este tipo de comunicación, los mensajes se envían y se reciben por medio de buzones. Un *buzón* es un objeto en donde los procesos colocan y eliminan

mensajes, en cada buzón se utiliza un identificador único; en este tipo de comunicación, un proceso se comunica con otro utilizando diferentes buzones. Dos procesos pueden comunicarse solamente si tienen un buzón compartido, por ejemplo:

- *send* (**A**, mensaje). Enviar un mensaje al buzón **A**.
- *receive* (**A**, mensaje). Recibir un mensaje del buzón **A**.

En este tipo de comunicación, se tienen las siguientes propiedades:

- Se establece un enlace entre un par de procesos, sólo si ambos tienen un buzón compartido.
- Un enlace se puede asociar con más de dos procesos.
- Entre cada par de procesos pueden existir varios enlaces diferentes, en donde a cada enlace le corresponde un buzón.

1.3.5. Interrupciones

Una interrupción es una señal que indica que se suspende la ejecución de un programa y activa un programa especial conocido como “manejador de interrupción”.

Excepciones de programa: son eventos que ocurren cuando se ejecuta un programa, y que requiere la ejecución de un fragmento de código que se encuentra fuera del flujo normal de ejecución. Son generadas por *software.* Las excepciones incluyen la ejecución de instrucciones ilegales, o la división por cero, las excepciones de *software* incluyen aquellas que son detectadas y notificadas por los procesos o por el sistema operativo. Cuando ocurre una excepción, el control es transferido al sistema operativo, que ejecuta la rutina de tratamiento de excepción correspondiente.

Las interrupciones se pueden generar por las siguientes causas:

→ Excepciones de programa. Hay determinadas causas que hacen que un programa presente un problema en su ejecución, por lo que deberá generarse una interrupción para que el sistema operativo trate esta causa. Por ejemplo: el desbordamiento en las operaciones aritméticas, la división por cero, el intento de ejecutar una operación con código de operación incorrecto o de direccionar una posición de memoria no permitida.

→ *Interrupciones de reloj*. Genera interrupciones cada cierto tiempo para asegurar que ningún proceso retenga completamente el uso del procesador.

→ *Interrupciones de entrada/salida (E/S).* Los controladores de los dispositivos de E/S necesitan interrumpir para indicar que han terminado una operación o conjunto de ellas.

→ *Excepciones*. Cuando se detecta de un error de paridad en la memoria o cuando existe un corte de corriente, se avisan mediante la correspondiente interrupción.

→ *Instrucciones de TRAP.* Estas instrucciones permiten que un programa genere una interrupción, se utilizan principalmente para solicitar los servicios del sistema operativo.

1.3.6. Interbloqueo de procesos

En los sistemas operativos multiusuario y multiproceso, es preciso retomar lo que se refiere al interbloqueo como sigue: “es el bloqueo permanente de un conjunto de

procesos o hilos de ejecución en un sistema concurrente que compiten por un número finito de recursos.

A diferencia de otros problemas de concurrencia de procesos, no existe una solución general para los interbloqueos. Todos los interbloqueos surgen de necesidades que no pueden ser satisfechas, por parte de dos o más procesos." (Meza, n.d: 39).

Por ejemplo:

Los procesos **A** y **B** requieren imprimir cada uno un archivo grande de una unidad de respaldo. El proceso **A** solicita permiso para utilizar la impresora, se le concede el permiso. En ese momento el proceso **B** solicita utilizar la unidad de respaldo y se le concede. El proceso **A** solicita la unidad de respaldo y es negada hasta que **B** la libere. En ese momento el proceso **B** solicita la impresora que está siendo utilizada por **A**. Los procesos se bloquean de manera permanente.

1.3.7. Algoritmos de administración y procesos

En cuanto al éxito de la planificación de la CPU, puede observarse que depende de la siguiente propiedad en los procesos: la ejecución de los procesos consta de un ciclo de ejecución de la CPU y espera de entrada y de salida.

La planificación tiene los siguientes objetivos:

-- Reparto equitativo del procesador.
-- Eficiencia, optimizar el uso del procesador (mantener ocupado 100% de tiempo de la unidad central de proceso (CPU)).
-- Tiempo de respuesta, minimizar el tiempo de respuesta al usuario.
-- Tiempo de regreso, minimizar el tiempo que deben esperar los usuarios por lotes (*batch*) para obtener sus resultados.
-- Rendimiento, maximizar el número de tareas procesadas por hora.

(Carretero, 2001: 104).

Existen diferentes tipos de algoritmos asociados a la unidad central de proceso, los cuales tienen diversas propiedades que favorecen a una clase de procesos sobre otros.

→ Algoritmos de planificación

De acuerdo con lo que dice Silberschatz (2002: 139-156) se plantean los siguientes algoritmos:

1. **Primero en llegar**, **primero en salir**. En este esquema se asigna el uso de CPU al primer proceso que lo solicite (*first come*, *first-served*, FCFS). La implementación de este tipo de política (FCFS) se maneja de manera sencilla con una cola tipo FIFO (*first input*, *first output*). Cuando un proceso entra a la cola de los procesos listos, su PCB se va al final de la cola. Cuando el CPU se encuentra libre, se asigna su uso al proceso que se encuentra al inicio de la cola.

2. **Primero el trabajo más cortó**. Este algoritmo (*shortest*-job-*first*, SJF) asocia la longitud de un proceso a la disposición (ráfaga) de la unidad central de proceso. Cuando el CPU está disponible, se le asigna al proceso que tiene la ráfaga más corta. Si dos procesos tienen la misma longitud de ráfaga, se utiliza la planificación FCFS del algoritmo anterior.

3. **Prioridad.** Una prioridad está asociada a cada proceso, y el CPU se asigna al proceso que tenga la prioridad más alta. Cuando los procesos tienen la misma prioridad se planifica de acuerdo a FCFS.

4. **Roud-robin.** Este algoritmo se denomina por turnos (*round-robin*) fue diseñado para sistemas de tiempo compartido. Es similar a FCFS sólo que agrega apropiación para conmutar entre procesos; es decir, define una unidad de tiempo conocido como *quantum;* es decir*,* una unidad de tiempo de 10 a 100 milisegundos. La cola de procesos listos es manejada como una cola de tipo circular. El planificador del CPU da vueltas sobre la cola de procesos listos y asigna el uso de CPU a cada proceso durante un intervalo de hasta 1 *quantum*.

5. **Colas de niveles múltiples**. Este algoritmo divide la cola de procesos listos en varias colas separadas. Los procesos se asignan a una cola de acuerdo a ciertas propiedades del proceso (tamaño de memoria, tipo de proceso, prioridad, etcétera). Se pueden tener varias colas y cada una tiene su propio algoritmo de planificación.
   Ejemplo: colas de procesos de niveles múltiples como son los procesos del sistema, procesos por lotes, procesos de estudiantes, etcétera.

6. **Colas de niveles múltiples y de retroalimentación.** Este tipo de algoritmo permite que un proceso se mueva entre diferentes colas; es decir, separa procesos con características diferentes de ráfaga del CPU. Si un proceso utiliza demasiado tiempo del uso del CPU, será movido a una cola de menor prioridad.

Para seleccionar el algoritmo a utilizar en una situación particular, se deben considerar las diferentes propiedades que tienen.

"Utilización de CPU: mantener la CPU tan ocupada como sea posible. La utilización puede estar entre 0 y 100 por ciento, de esto depende que el uso del sistema sea ligero (40%) o pesado (hasta 90%).

* *Rendimiento* (*throughput):* se mide por el número de procesos que se terminan por unidad de tiempo. En el caso de procesos de larga duración, esta tasa podría ser un proceso por hora, para transacciones breves el rendimiento podría ser de 10 procesos por segundo.

* *Tiempo de entrega*: tiempo que se requiere para ejecutar un proceso, desde el momento en que se presenta un proceso hasta su terminación. El tiempo de entrega, es la suma de los periodos que se consumen esperando llegar a la memoria, esperando en la cola de listos, en ejecución en el CPU y realizando operaciones de E/S.

* *Tiempo de espera*: el tiempo de espera es la suma de los periodos esperando en la cola de listas.

* *Tiempo de respuesta*: esta medida es el tiempo que requiere para empezar a responder y no cuánto se requiere para producir la salida de dicha respuesta. El tiempo de entrega por lo general está limitado por la velocidad del dispositivo de salida.” (Meza, n.d: 47)

1.4. Administración de memoria

1.4.1. Administración de la memoria

La administración de la memoria se refiere a los métodos y operaciones para obtener la máxima utilidad de ésta a través de la organización de procesos y programas que se ejecutan en una computadora.

La parte del sistema operativo que administra la jerarquía de memoria se llama administrador de memoria.

> Los sistemas de administración de memoria se dividen en dos: los que traen y llevan procesos entre la memoria principal y el disco duro durante la ejecución (intercambio y paginación), y los que no lo hacen.
>
> La memoria consiste en un arreglo de bytes, cada uno con su propia dirección. La unidad central de proceso (CPU) acude por instrucciones a la memoria de acuerdo con el valor del contador de programa. (Meza, n.d: 55).

→ Esquemas de administración de memoria

1. Multiprogramación sin intercambio ni paginación. Es el esquema más sencillo de la administración de la memoria, ya no tiene aplicación. Consistía en que al ejecutar un programa, se utilizaban todos los recursos de la computadora (memoria, *software*, etcétera).
2. Multiprogramación con particiones fijas. La multiprogramación eleva el aprovechamiento del CPU. Esto se logra dividiendo la memoria en particiones. (…)

3. *Modelado de la multiprogramación*. Un proceso pasa una fracción p de su tiempo esperando a que terminen operaciones de E/S. Si hay n procesos en la memoria a la vez, la probabilidad de que todos estén esperando E/S (el CPU estará inactivo) es p^n. Entonces el aprovechamiento de la CPU está dado por la fórmula:

Aprovechamiento de $CPU = 1 - p^n$.

1.4.2. Participaciones fijas y dinámicas

Generalmente, la memoria se divide en dos particiones: una que es para el sistema operativo residente y otra para los procesos que tienen que ver con los usuarios.

Uno de los métodos más sencillos para la asignación de memoria consiste en dividirla en un número de particiones de tamaño *fijo*. Cada una de estas particiones puede contener exactamente un proceso. De tal modo que el grado de multiprogramación se limita por el número de particiones. Cuando una partición está disponible, se selecciona un proceso de la cola de entrada y se carga en ella y al terminar dicho proceso, la partición queda disponible para otro. Este método fue empleado originalmente por el sistema operativo OS/360 de IB (denominado MFT); en la actualidad ya no se usa.

El método de particiones *dinámicas* utiliza toda la memoria al cargar las primeras tareas en el sistema que no son del mismo tamaño de las que acaban de salir de la memoria, y se acomodan en los espacios disponibles de acuerdo a su prioridad.

1.4.3. Asignación estática de la memoria

La asignación estática de memoria consiste en el proceso de asignar memoria en tiempo de compilación de un programa, en donde los objetos están vigentes desde que inicia la ejecución del programa hasta que termina. La asignación de este tipo de memoria se define, dentro de la sintaxis de los lenguajes de programación, de tal manera que se pueden reservar *bytes* de memoria que ningún otro programa pueda utilizar. Los objetos que se pueden manejar para este tipo de memoria son: variables estáticas, variables globales, literales, etcétera.

1.4.4. Asignación dinámica de la memoria

Tiene que ver con la asignación de almacenamiento de memoria para su utilización por un programa de cómputo durante el tiempo de ejecución; es decir, a medida que se va requiriendo espacio, el programa va solicitando más memoria al sistema operativo. En este tipo de asignación de memoria, los datos se crean y se destruyen mientras se ejecuta el programa, de tal manera que se evita el desperdicio de la misma.

1.4.5. Paginación

De acuerdo con los planteamientos de Silbertschatz (2002: 269-274), la paginación es un diseño que permite que el espacio de direcciones lógicas de un proceso no sea contiguo y evita el problema de ajustar las porciones de memoria de tamaño variable en el almacén de respaldo (del cual sufrían la mayoría de los diseños anteriores de administración de la memoria).

→**Algoritmos para reemplazo de páginas** (Tanenbaum, 2003: 214-227).

Cuando se presenta una falla de página, el sistema operativo tiene que escoger la página que desalojará de la memoria para hacer espacio para colocar la página que traerá del disco. Si la página a desalojar fue modificada mientras estaba en la memoria, deberá reescribirse en el disco para actualizar la copia. En cambio, si la página no se ha modificado (por ejemplo, si contiene texto de programa), la copia en disco ya estará actualizada y no será necesario reescribirla. La nueva página simplemente sobrescribe la que está desalojando. Aunque sería posible escoger una página al azar para desalojarla cuando haya una falla de página, el desempeño del sistema mejora mucho, si se escoge una página que no se usa mucho. Si se desaloja una página muy utilizada, lo más seguro es que pronto tenga que volver a traerse a la memoria, con el consiguiente gasto extra.
A continuación se describen los diferentes algoritmos para el reemplazo de páginas:

1. Algoritmo óptimo de reemplazo de páginas. Este algoritmo indica qué página debe desalojarse de la memoria cuando existe una falla de página.

2. Algoritmo de reemplazo de páginas no utilizadas recientemente. Las computadoras con memoria virtual asocian a cada página dos estados de bits. **R** se enciende cada vez que se hace referencia a la página para leer y escribir. **M** se enciende cada vez que escribe en la página (se modifica). Cuando se presenta una falla de página, el sistema operativo examina la totalidad de páginas y las divide en cuatro categorías con base en los valores actuales de **R** y **M**.

- Clase 0: no solicitada, no modificada.
- Clase 1: no solicitada, modificada.
- Clase 2: solicitada, no modificada.
- Clase 3: solicitada, modificada.

3. Algoritmo de reemplazo de páginas, primero en entrar, primero en salir. En este algoritmo (FIFO; *first-in*, *first-out*) el sistema operativo mantiene una lista de todas las páginas que actualmente están en la memoria, con la más antigua al principio de la lista y la más reciente al final. Al presentarse la falla de página, se desaloja la página que está al principio y la nueva se deja al final.

4. Algoritmo de reemplazo de página de segunda oportunidad. Este algoritmo es una modificación de FIFO que evita desalojar una página que se utiliza mucho. Consiste en examinar el *bit* R de la página más antigua. Si el valor de *bit* es 0, indica que la página no sólo es antigua; sino que también se usa mucho. Por lo que es reemplazada de inmediato. Si el valor del *bit* R es 1, entonces se apaga y la página se coloca al final de la lista de páginas y su tiempo de carga se actualiza como si acabara de llegar a la memoria.

5. Algoritmo de páginas tipo reloj. Este algoritmo mantiene las páginas en una lista circular (tipo reloj). Cuando se presenta una falla en la página, se examina la que apunta la manecilla. Si el *bit* es 0 se desaloja, la nueva página se inserta en su lugar y la manecilla se adelanta un poco. Si **R** es 1, se cambia a 0 y la manecilla se adelanta a la siguiente. El proceso se repite hasta encontrar una con **R** = 0.

6. Algoritmo de reemplazo de página menos recientemente utilizada. Este algoritmo denominado (LRU; *least recently used*) considera que cuando se presenta una falla de página, se desaloja la que tiene más tiempo sin utilizarse. Una de las dificultades para su implementación radica en que la lista debe actualizarse cada que se hace referencia a la memoria. Encontrar una lista, borrarla y reinsertarla es una operación muy tardada y tiene un alto costo en memoria y *hardware*.

7. Algoritmo de reemplazo de páginas de conjunto de trabajo. En este algoritmo, los procesos inician sin tener páginas en la memoria. Cuando el CPU

trata de obtener la primera instrucción, habrá una falla de página y el sistema operativo traerá a disco la página que tiene la primera instrucción. Después de cierto tiempo, el proceso tiene casi todas las páginas que necesita y su ejecución se estabiliza. A esto se le conoce como **paginación por demanda**, ya que las páginas se cargan sólo cuando se necesitan y no por adelantado.

8. Algoritmo de reemplazo de páginas WSClock. Tiene como base el algoritmo de reloj; pero a la vez utiliza la información de conjunto de trabajo. En principio, la lista está vacía y cuando se carga la primera página, ésta se añade a la lista. En la medida que se accede a las páginas, se incorporan a la lista formando un anillo. Al igual que en el algoritmo de reloj, cada vez que hay una falla de página se examina primero la página que apunta a la manecilla. Si el *bit* **R** es 1 significa que la página se utilizó durante el tiempo actual, por lo que la página no es candidata idónea para desalojarse, por lo que se apaga el *bit* **R**, se adelanta la manecilla a la siguiente página y se repite el algoritmo. Este algoritmo es fácil de implementar y tiene un buen desempeño.

1.4.6. Políticas de reemplazo de páginas

Una de las políticas principales de un sistema de memoria virtual y que puede afectar el rendimiento, es la que determina qué página reemplazar cuando hay que traer otra de la memoria secundaria y no existe espacio libre. Aunque gestionar la transferencia de información entre la memoria principal y el disco duro no se limita al sistema de memoria virtual; sino que también se da en la gestión de la memoria *caché* del sistema de archivos y en los sistemas manejadores de bases de datos. Muchos algoritmos utilizados en estos ámbitos son similares a los utilizados en la memoria virtual, hay que considerar que existe una diferencia importante en el modo de operación del sistema de archivos o de un manejador de bases de datos y el sistema de memoria. En los primeros, existe una petición explícita por parte de

la aplicación para acceder a la información, pudiendo el sistema operativo o el manejador de bases de datos tener el control sobre los accesos que se van realizando. En cambio, en la memoria, los procesos acceden directamente a la misma sin que el sistema operativo tenga control de ello y sólo lo toma cuando se produce una falla de página. Las políticas de reemplazo se clasifican en dos categorías: *reemplazo global y local*. En el reemplazo global se puede seleccionar para satisfacer la falla de página de un proceso un marco que tenga actualmente asociada una página de otro proceso; es decir, un proceso puede quitarle un marco de página a otro. En el reemplazo local requiere que para servir la falla de página de un proceso sólo puedan utilizarse marcos de páginas libres o marcos ya asociadas al proceso. El objetivo básico de cualquier algoritmo de reemplazo es minimizar la tasa de fallas de página y son los descritos anteriormente.

1.4.7. Memoria virtual

La memoria virtual es una técnica de administración de la memoria, utilizada por el sistema operativo, que permite que se ejecuten procesos que pueden no estar completamente en la memoria principal. Este tipo de memoria está formada por circuitos integrados de dos tipos: ROM (*Ready Only Memory*) “Memoria de sólo lectura” y RAM (*Random Access Memory*) “Memoria de acceso aleatorio”:

> La memoria virtual es la separación de la memoria lógica del usuario y de la memoria física [del equipo]. Esta separación permite ofrecer a los programadores una memoria virtual grande cuando sólo se dispone de una memoria física pequeña.[7]

Por ejemplo, un programa de 16 MB puede ejecutarse en una computadora de 4 MB si se elige con cuidado cuales 4 MB se tendrán en la memoria en cada instante,

[7] Véase, Universidad de Jaén, Departamento de Informática: “Tema 7: Memoria virtual; 7.1.” material electrónico disponible en: http://wwwdi.ujaen.es/~lina/TemasSO/MEMORIAVIRTUAL/1y2Motivaciones,ventajasyEstrategiasdeadministracion.htm Fecha de recuperación: 08 de enero de 2009.

intercambiando fragmentos del programa entre el disco y la memoria, según se requieren.

1.4.8. Memoria escondida (*caché*)

El término memoria *caché* se aplica normalmente a una memoria más pequeña y más rápida que la memoria principal, se sitúa entre la memoria RAM y el CPU. Es un tipo de memoria RAM estática (SRAM) y está diseñada para contener información que se utiliza con mucha *frecuencia* en un proceso, con el objetivo de evitar accesos a otras memorias (principal), reduciendo de manera considerable el tiempo de acceso, ya que es más rápida que la memoria principal. Por ejemplo, cuando el procesador lee los datos y se almacenan en la memoria principal, los datos también se almacenan en la memoria *caché*. Si el procesador los requiere de nuevo, los lee de la memoria *caché* y no de la memoria principal y se incrementa la velocidad. En una computadora la memoria *caché* es menor que la principal y su costo es alto.

RESUMEN DE LA UNIDAD

Un sistema de cómputo está formado por los siguientes elementos: el *hardware*, el sistema operativo, los programas de sistema y los usuarios. El sistema operativo es un *software* que le da significado y orden a las aplicaciones para interactuar con el *hardware* de una computadora y administra cuatros aspectos importantes: el *hardware*, el *software*, la memoria y los datos. Los sistemas operativos se han desarrollado a través del tiempo logrando un mejor desempeño del sistema de cómputo y proporcionando mejores plataformas para el desarrollo y ejecución de programas. Los sistemas por lotes permitieron la ejecución secuencial de trabajos y mejoraron el uso de la computadora. Los sistemas de tiempo compartido introdujeron la multiprogramación que permite que varios trabajos se mantengan en memoria, lo que mejoró el uso del CPU y tiempo de ejecución de los trabajos. La multiprogramación también permitió que los sistemas de tiempo compartido sean utilizados de manera simultánea por cientos de usuarios al mismo tiempo. Los sistemas operativos para PC aprovecharon el desarrollo tecnológico para que una sola persona utilice de manera exclusiva el CPU, *software*, etc. Los sistemas paralelos tienen más de un CPU en estrecha comunicación. Los de tiempo real se utilizan como dispositivos de control en aplicaciones dedicadas. El desarrollo de internet, las comunicaciones, los lenguajes de programación, la electrónica, etc., han impulsado los sistemas distribuidos mejorando el cómputo y la reducción de costos. Los conceptos fundamentales aplicados a los sistemas operativos multiusuario son principalmente los procesos y la administración de la memoria.

Un proceso se puede considerar como un programa en ejecución, para realizar su tarea requieren de tiempo de CPU, memoria, archivos y dispositivos de E/S, los cuales se asignan cuando se crea el proceso o cuando está en ejecución. Cuando

se ejecuta un proceso, cambia el estado de su actividad y puede ser: **nuevo,** cuando el proceso se está generando; en **ejecución,** se están ejecutando instrucciones; en **espera,** el proceso está esperando a que ocurra algún evento; **listo,** el proceso está en espera de ser asignado a un procesador y **terminado** cuando el proceso ha terminado su ejecución. Los procesos se identifican por medio de un identificador de proceso que puede ser proceso creador o padre y proceso nuevo o hijo. Cada proceso se representa en el sistema operativo por medio del bloque de control del proceso (PCB) que contiene información referente a **estado del proceso**, contador de programa, registros de CPU, información sobre la planificación del CPU y de la administración de la memoria, información de E/S, etc.

La administración de la memoria se refiere a los métodos y operaciones para obtener la máxima utilidad de ésta a través de la organización de procesos y programas que se ejecutan en una computadora. La parte del sistema operativo que administra la jerarquía de memoria se llama *administrador de memoria*, se dividen en dos categorías: los que ejecutan procesos entre la memoria principal y el disco duro y los que no realizan este esquema. La memoria consiste en un arreglo de **bytes**, cada uno con su propia dirección. La unidad central de proceso (CPU) acude por instrucciones a la memoria de acuerdo con el valor del contador de programa. Los esquemas de administración de la memoria son: multiprogramación sin intercambio de paginación; multiprogramación con particiones fijas y el modelado de la multiprogramación. Uno de los aspectos más importantes en la administración de la memoria es la paginación, ya que permite que el espacio de direcciones lógicas de un proceso no sea contiguo y evita el problema de ajustar las porciones de memoria de tamaño variable en el almacén de respaldo (como la mayoría de los diseños anteriormente desarrollados). Existen diferentes algoritmos para el reemplazo de páginas, cada uno tiene sus propias características y son: algoritmos óptimo; reemplazo de páginas no utilizadas recientemente; primero en entrar primero en salir (FIFO); segunda oportunidad; de tipo reloj; menos recientemente utilizada; conjunto de trabajo y WSClock.

Los que logran mejor desempeño en la paginación y se pueden implementar con eficiencia son: WSClock y el de reemplazo de página menos recientemente utilizada, ambos están basados en LRU (*Least recently used*).

BIBLIOGRAFÍA

SUGERIDA

Autor	Capítulo	Páginas
Silberschatz, Abraham.	4 9	87-114 255-296
Tanenbaum, Andrew S.	1 4	1-20 214-228
Carretero, Jesús.	3 4	77-93 187-208

Licenciatura: Informática

UNIDAD 2

WINDOWS SERVER

SUAyED UNAM
Sistema Universidad Abierta y Educación a Distancia

Plan 2012 2016 actualizado

OBJETIVO PARTICULAR

El alumno conocerá las características principales, operación y capacidad del sistema operativo Windows Server.

TEMARIO DETALLADO

(8 horas)

2. Windows Server

2.1. Características generales
2.2. Arquitectura de Windows Server
2.3. Active Directory
2.4. Sistema de archivos

INTRODUCCIÓN

Es indudable que las características y necesidades de las organizaciones en cuanto a la administración de sus servicios y recursos, formas de comunicarse con los empleados, compartir información en red, etcétera, requieren del uso de nuevas tecnologías de información para responder efectivamente a sus necesidades, por lo que se necesita mayor capacidad de cómputo.

Microsoft[MR] maneja diversas soluciones de sistemas que ayudan a atender ciertas necesidades tecnológicas personales y empresariales, entre ellas están las versiones dirigidas al almacenamiento y administración de la información (como SQL, Visual Basic), las que se usan para el desarrollo de aplicaciones (como Visual Studio, Framework), las orientadas a la productividad en los negocios (*Exchange Server, Office Communications Server*), las que ayudan a administrar los recursos de infraestructura y *software* en nivel empresarial (*Windows Server*), entre otros.

En la década de los noventa, esta empresa puso a disponibilidad de los usuarios el cómputo tipo cliente/servidor a diversas escalas. Windows Server fue la solución, un sistema operativo para funciones de servidor de archivos, impresión, aplicaciones y servicios Web. Hoy ha agregado diversas características que simplifican las tareas del usuario, mejoran la administración de dispositivos, proveen interfaces de usuario simplificadas, emplean tecnología NT (Nueva Tecnología), contiene capacidades *plug and play*, entre otros. También cuenta con un conjunto completo de servicios de infraestructura basados en *Active Directory*, que centralizan la gestión de usuarios, grupos, servicios de seguridad y recursos de la red.

Se han venido abordando diferentes versiones de Windows Server, desde la 2000, 2003, 2008, 2012 y actualmente ya se habla de la versión 2013. Es decir, es una plataforma que va creciendo con base en las necesidades de la organización, donde se pueden crear aplicaciones modernas y asegurar que los usuarios puedan acceder a los datos y servicios en cualquier lugar. Éste, a fin de cuentas, es un sistema operativo que hace lo que todos los sistemas, administrar el *hardware* y ser una plataforma para las aplicaciones, pero también amplía sus horizontes y servicios para las necesidades de los usuarios que navegan, colaboran y se comunican en la nube, así como para los desarrolladores y personas involucradas con las TI.

En la presente unidad, abordaremos las características más importantes de este sistema operativo, su arquitectura, fortaleza del directorio activo y su sistema de archivos propio de esta tecnología, con el fin de que el alumno lo conozca y tenga elementos para decidir en qué aplicaciones sería conveniente utilizarlo.

2.1. Características generales

Windows Server es un sistema operativo con varios propósitos para el manejo de la información. Su desarrollo tecnológico y gran variedad de productos adicionales de *software* permiten que su utilización se adapte a empresas pequeñas o grandes en cualquier parte de mundo. Se utiliza por medio de licencias de *software* que se adquieren de acuerdo a las necesidades específicas de las empresas, lo que reduce el costo e impulsa el desarrollo de las empresas. A cada versión nueva, se van incorporando características y mejoras nuevas; incluyendo la virtualización, almacenamiento, redes, computación en internet, automatización, entre otras aplicaciones.

Windows Server cuenta con varias versiones que se caracterizan por los entornos en los que se aplica, algunas de estas versiones son las siguientes:

- **Windows Server 2012 Datacenter.** Diseñado para aplicaciones que requieran entornos virtuales, alta disponibilidad, aplicaciones basadas en la nube y agrupación de *clusters*; es decir, en un solo servidor físico se puede ejecutar sin límite el uso de máquinas virtuales, lo que reduce significativamente costos en *hardware* y licencias. El manejo de usuarios es ilimitado.

- **Windows Server 2012 Standard.** Diseñado para entornos que requieren un servidor físico y hasta 2 máquinas virtuales. El manejo de usuarios es ilimitado.

- **Windows Server 2012 Essentials.** Ideal para pequeñas empresas que tienen un máximo de 25 usuarios simultáneos y 50 dispositivos. Si se requieren conectar más cantidad de usuarios, se tendrán que adquirir el licenciamiento correspondiente.

- Windows Server 2012 Foundation. Ideal para empezar a trabajar con un equipo con funciones de servidor. Windows Server soporta hasta 15 usuarios simultáneos**.** Si se requieren conectar más cantidad de usuarios se tendrá que adquirir el licenciamiento correspondiente.

Windows Server[8] se caracteriza, esencialmente, por los siguientes aspectos:

- → **Menor costo total del uso de licencias.** Reduce el costo de organizar y ejecutar una red de computadoras al permitir la instalación automática y actualización de aplicaciones. Esto facilita la instalación y configuración de las computadoras cliente de manera remota sin necesidad de que el administrador asista de manera física al sitio de instalación.
- → **Seguridad.** Identifica a los usuarios antes de proporcionar acceso a los recursos de un equipo o red; datos, archivos, directorios, impresoras, etcétera. Soporta el protocolo Kerberos y seguridad de infraestructura de llave pública (*Public Key Infraestructure*, PKI).
- → **Servicios de directorio.** Almacena y gestiona información de los servicios de *Active Directory* en el directorio, el cual es la base de datos que almacena información acerca de los recursos de la red.
- → **Rendimiento y dimensionabilidad.** Soporta múltiples procesadores (dependiendo de la versión), que se configuran normalmente como un servidor de archivos e impresoras.
- → **Servicios de red y comunicación.** Soporta los protocolos de comunicación de red más comunes: TCP/IP y IPX/SPX. También proporciona conectividad con los sistemas: *Novell Netware*, *Unix* y *Apple Talk*.
- → **Integración con internet.** Integra los escritorios de los equipos de los usuarios con internet para que puedan acceder a los recursos por medio de la red local, Intranet e Internet. Incluye *Internet Information Server* (IIS) de

[8] Microsoft Corporation (2001). Microsoft Windows 2000 Server. Madrid: Mc Graw Hill, pp. 4-5.

Microsoft como plataforma segura del servidor Web, utilizado para servir de anfitrión a los sitios web de internet e intranet sobre los servidores de red.

- **Herramientas de seguridad integradas.** Proporciona los medios para crear herramientas personalizadas para gestionar computadoras locales y remotas con una única interfaz estándar. Por ejemplo administrar en una sola aplicación todas las impresoras remotas y locales.
- **Soporte de *hardware*.** Soporta *hardware plug and play*, instala y configura de manera automática cualquier dispositivo; impresoras, discos y periféricos diversos.
- **Sistema de archivos.** Utiliza el sistema de archivos NTFS que proporciona:
  - Recuperación del sistema de archivos.
  - Grandes particiones en discos duros.
  - Seguridad del sistema de archivos.
  - Compresión de información.
  - *Disk quotas* para determinar el espacio asignado a los usuarios de acuerdo a sus necesidades.
- **Quality of Service (QoS).** Proporciona calidad del servicio a determinados tipos de tráfico de voz, video y datos, que así lo requieran. Por ejemplo: se puede dar prioridad y garantizar un ancho de banda mínimo de los servicios de voz sobre los de datos o viceversa. Algunos servicios de voz como VoIP (transmisión de voz sobre el protocolo IP) requieren calidad de servicio para que la transmisión sea adecuada al tipo de servicio. La transmisión debe ser correcta de emisor a receptor (*end to end*).
- **Clustering.** Windows Server permite agrupar un grupo de computadoras independientes para que ejecuten aplicaciones de *software* de misión crítica, Web, bases de datos de alto rendimiento, etcétera.

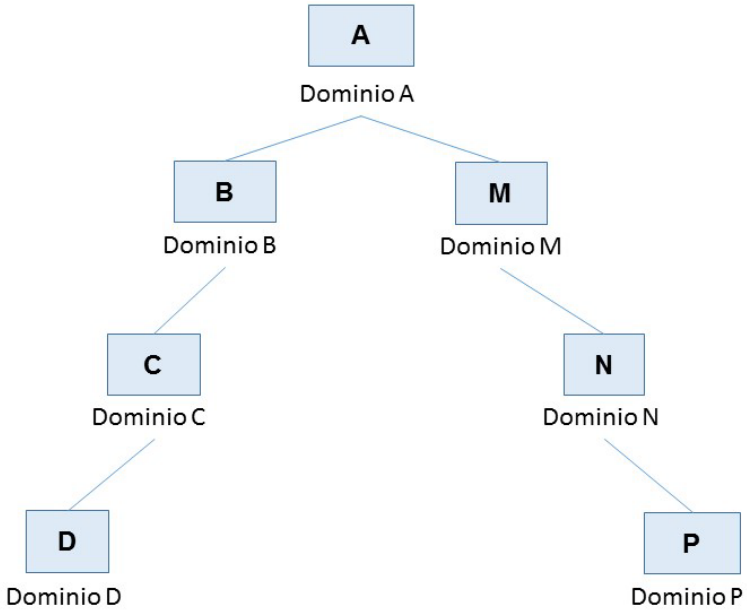
  Utiliza los recursos de cada equipo (procesador, memoria, red, etcétera) de manera transparente al usuario; es decir, el usuario no identifica que las aplicaciones o servicios que tiene utilizan la tecnología *cluster*. Su uso

mejora el rendimiento, la disponibilidad de las aplicaciones y evita la compra de grandes equipos.

- **Servicio terminal.** Éste provee de acceso remoto al escritorio del servidor a través de un emulador de terminal; es decir, el administrador del servidor puede realizar ciertas tareas sin utilizar el entorno gráfico, aumentando el tiempo de respuesta, sobre todo en entornos remotos.
- **Servicios de instalación remota.** El servicio de instalación remota (RIS) proporciona al administrador la posibilidad de utilizar un sistema operativo desde fuera de la organización, sin la necesidad de visitar físicamente cada computadora del usuario.
- **Dominios.** Un dominio es un grupo de computadoras conectadas en red y que comparten un área común para almacenar información segura. Un dominio provee de una administración centralizada y fiable de los recursos de la red. Los usuarios en una computadora pueden acceder a los recursos compartidos en otras computadoras en el dominio con los permisos apropiados.

Los dominios se parecen a los grupos de trabajo y ofrecen las siguientes ventajas adicionales:

- Un único proceso de acceso al equipo.
- Una sola cuenta de usuario.
- Administración centralizada.
- Escalabilidad para redes grandes.



Dominios Windows Server.

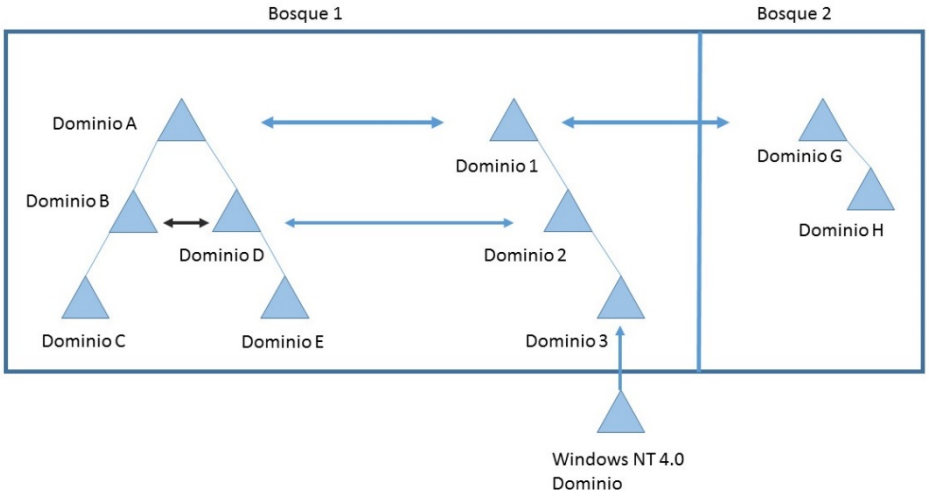
Elaboración con base en http://support.microsoft.com/kb/310997/es, 29/04/2016

Existen tres formas de organizar los dominios:

- **Controlador de dominio**: consiste en una computadora que ejecuta Windows Server y administra cada dominio.
- ***Trees* (del inglés, árboles):** arreglo jerárquico de dominios Windows Server que comparten un nombre en común.
- ***Forest* (por su traducción del inglés: bosque):** grupo de *trees* que no comparten el nombre común, pero comparten una configuración común.

- ***Active Directory*.** Es un servicio de directorios que almacena la información acerca de los objetos de la red y provee una estructura jerárquica que hace más fácil organizar los dominios y los recursos; es decir, organiza los directorios en secciones para almacenar un número muy grande de objetos, provee un repositorio central para brindar y distribuir información acerca de los objetos de la red incluyendo usuarios, grupos, impresoras, etcétera y

finalmente provee seguridad integrada a través del proceso de autentificación y el control de acceso a los objetos de la red. En el siguiente esquema se muestra un diagrama de organización y controlador de dominios en Windows Server.



Organización de dominios Windows Server

Elaboración con base en:

http://www.flashcardmachine.com/windows-server-20082008r2.html

2.2. Arquitectura de Windows Server

Windows Server[9] es un sistema operativo para funciones de servidor, su diseño es modular, a base de pequeños componentes de *software* independientes y que trabajan juntos para realizar las tareas de un sistema operativo. Cada componente proporciona un conjunto de funciones que actúan como una interfaz para el resto del sistema.

A continuación se describen los principales componentes de la arquitectura de Windows Server.

Modo usuario La capa de modo usuario de Windows Server está formada por un conjunto de componentes denominados subsistemas. Un subsistema pasa peticiones de E/S al controlador del modo núcleo apropiado a través de los servicios de los sistemas de E/S. Los subsistemas son transparentes a los usuarios, ya que evitan que sus usuarios finales y aplicaciones tengan que conocer el funcionamiento de los componentes.
Subsistemas de entorno Los subsistemas de entorno permiten a Windows Server ejecutar aplicaciones escritas para diferentes sistemas operativos a través de la interfaz de programación de aplicaciones (API). Los subsistemas de entorno aceptan las llamadas a API realizadas por las aplicaciones y convierte las llamadas API en un formato entendible por Windows Server. Los subsistemas de entorno son:

[9] Microsoft Corporation (2001). *Microsoft Windows 2000 Server*. Madrid: McGraw Hill, p. 6-16.

- Win32 el cual controla las aplicaciones basadas en Win32 y proporciona un ambiente para aplicaciones basadas en Win16 y Microsoft MS-DOS.
- POSIX que proporciona API para aplicaciones basadas en POSIX.

POSIX es una norma de interfaz de sistemas operativos transportables desarrollado por el *Institute of Electrical and Electronics Engineers* (IEEE) para asegurar la transportabilidad entre aplicaciones de diferentes sistemas operativos. Las aplicaciones que se ejecutan con ellas no tienen acceso al *hardware* ni a los controladores del dispositivo, están limitadas a un espacio de direcciones asignado. Los subsistemas de entorno están forzados a utilizar memoria del disco duro (memoria virtual), también estos subsistemas se ejecutan con una prioridad menor que los procesos en modo núcleo, por lo que sus procesos tienen menor acceso al uso del CPU que los de modo núcleo.

Subsistemas integrales

Los subsistemas integrales realizan funciones esenciales del sistema operativo tales como:

- Seguridad: crea *tokens* (ver definición en el glosario) de seguridad y detecta derechos y permisos asociados con cuentas de usuario.
- Servicio de estación de trabajo: es un subsistema integral de red que proporciona una API (Interfaz de Programación de Aplicaciones) para acceder al reexpedidor de red.
- Servicio de servidor: es un subsistema integral de la red que proporciona una API para acceder al servidor de la red. El servicio de servidor permite a una computadora con Windows Server acceder a los recursos de la red.

Modo núcleo

La capa modo núcleo de la arquitectura de Windows Server tiene acceso al sistema de datos y *hardware*. El modo núcleo proporciona un acceso a la memoria y se ejecuta en un área de memoria protegida. Determina cuándo una secuencia particular de código se ejecuta mediante un criterio de prioridades ya que cada hilo de ejecución o subproceso tiene asociado un atributo de prioridad.

El modo núcleo también proporciona prioridad a interrupciones de *hardware* y *software* de forma que parte del código de modo núcleo se ejecuta en mayores niveles de petición de interrupción (IRQL, *Interrupt Request Levels*).

Windows Executive

Executive realiza la mayor parte de la administración de E/S y de objetos, incluyendo la seguridad. Varios componentes de *Executive*, tales como el administrador de memoria virtual (VMM, *Virtual Memory Manager*) y el administrador de E/S, definen uno o más tipos de objetos. Estos componentes proporcionan servicios de sistema y programas internos.

Los servicios de sistema están disponibles tanto para los subsistemas de modo usuario como para otros componentes *Executive*. A continuación se mencionan algunos componentes del modo núcleo contenidos en *Executive*.

- Administrador de E/S. Proporciona servicios fundamentales para controladores de dispositivo y traduce comandos de lectura y escritura en modo usuario IRP (*Input-Output Request Packet*). El administrador de E/S gestiona la entrada y el reparto de la salida a diferentes dispositivos.
- Administrador de comunicación entre procesos (*Interprocess Comunication, IPC Manager*). Gestiona la comunicación entre clientes y servidores, el administrador IPC gestiona la comunicación entre los subsistemas de entorno y *Executive*. El subsistema actúa como un cliente solicitando información y *Executive* actúa como un servidor para satisfacer la solicitud de información.
- Administrador de memoria virtual (VMM, *Virtual Memory Manager*). Implementa y controla la memoria virtual, una gestión de memoria que proporciona un espacio de direcciones. El VMM permite al sistema operativo utilizar el almacenamiento en discos duros externos (periféricos) como si fuera parte de la memoria física. La memoria virtual utiliza la memoria física y el almacenamiento en discos duros y controla los requerimientos de paginación, lo que le permite la utilización del espacio

del disco como un área de almacenamiento para mover el código y los datos dentro y fuera de la memoria física RAM.

- Administrador de procesos. Crea y termina procesos y hebras. También suspende y reanuda hebras y recupera información de procesos y hebras. Una hebra es un conjunto específico de comandos dentro de un programa.
- Administrador *Plug and Play* (PnP). El administrador PnP soporta las actividades *Plug and Play* en tiempo de inicio, tiene interfaz con: HAL, *Executive* y con los controladores de dispositivos. Mantiene un control centralizado, dirigiendo a los controladores del *bus* a que ejecuten la enumeración, configuración e inicio de dispositivos en modo usuario. Puede detener o eliminar dispositivos, según se requiera.
- Administrador de ventanas e interfaz de dispositivo gráfico (GDI). Administra el sistema de pantalla a través de Win32k.sys y realiza las siguientes funciones:

  - Administrador de ventas. Controla la visualización de ventanas y gestiona la salida por pantalla.
  - GDI. Contiene las funciones necesarias para dibujar y manipular gráficos.

Capa de abstracción de *hardware* (HAL)

HAL es una interfaz entre el *hardware* y el sistema operativo, es responsable de proteger el sistema de las especificaciones del *hardware* como los controladores de interrupción e interfaces de entrada/ salida. Esta abstracción permite a Windows Server ser más portable, ya que el resto del sistema no se preocupa sobre qué plataforma de *hardware* se está corriendo el sistema operativo. HAL contiene:

- Código específico de *hardware* que gestiona las interfaces de E/S.
- Controladores de interrupción.
- Mecanismos de comunicación multiprocesador.

La capa HAL se diseñó para que Windows Server pueda ejecutarse en plataformas basadas en procesadores “Intel” o en otras, sin tener que mantener dos versiones separadas de *Windows Server Executive*. HAL se implementa

como una biblioteca de enlace dinámico y es responsable de todos los niveles de *hardware* para cada plataforma específica. HAL exporta rutinas de compatibilidad y esconde detalles de *hardware* específicos de cada plataforma sobre memoria *caché*, buses E/S y controladores de interrupción. También proporciona una interfaz entre el *hardware* de la plataforma y los componentes de *software* del sistema.

2.3. Active Directory

De acuerdo con Robert William (William, 2004: 201-230), un directorio es una colección de datos que guarda información sobre objetos que están relacionados entre sí; por ejemplo: un directorio telefónico, en el que se almacenan los nombres, direcciones y números de teléfonos de personas y empresas. De acuerdo con Microsoft, el directorio activo o *Active Directory* es una base de datos distribuida que permite almacenar información relativa a los recursos de una red con el objetivo de facilitar su localización y administración. Existen los términos *directorio* y *servicios de directorio*. El directorio es una base de datos de objetos de red, a la que se puede hacer referencia de varias maneras diferentes y almacena información relacionada con los recursos de la red para facilitar la localización y administración de los mismos. Un servicio de directorio se refiere a que la fuente de la información del directorio así como los servicios, hacen que la información esté disponible para los usuarios.

El servicio de *Active Directory* (Directorio activo), provee la estructura y funciones para la organización, administración y control de los recursos de la red de Windows Server e integra el concepto de internet de espacio de nombres con el servicio de directorios del sistema operativo, un servicio de directorio puede realizar las siguientes funciones:

- Dividir un directorio en varios almacenes que están ubicados en distintas computadoras de la red, esto permite que exista más espacio disponible al directorio y permite almacenar un número grande de objetos. Cuanto más grande sea una red, mayor número de recursos es posible gestionar.

- Replicar un directorio a otros equipos en la red para hacerlo disponible a más usuarios y hacerlo tolerante a fallas.
- Fortalecer la seguridad protegiendo a los objetos en su base de datos de intrusos interiores y exteriores que no tengan permiso de acceder a los objetos del directorio.

Los servicios de *Active Directory* integran el concepto de internet de espacios de nombres con los servicios de directorio de Windows Server. Esta integración permite la unificación y la gestión de múltiples espacios de nombres que existen en los entornos de *hardware* y *software* de redes corporativas. Los servicios de *Active directory* utilizan el sistema de nombres de dominio (*Domain Name System*, DNS) para su sistema de nombres y puede intercambiar información con cualquier aplicación o directorio que utilice el protocolo de acceso ligero (*Lightweight Directory Access Protocol*, LDAP). LDAP fue desarrollado como una alternativa más simple al protocolo de acceso a directorios X.500 (*Directory Access Protoco*l, DAP). X.500 es un conjunto de estándares que definen un servicio de directorio distribuido, desarrollado por la organización internacional de normalización (*International Standars Organization*, ISO). Los servicios de *Active Directory* soportan diferentes versiones de LDAP para intercambiar información entre directorios y aplicaciones.

La integración del DNS (*Domain Name System.* Sistema de Nombres de Dominio) con el *Active Directory* habilita a los corporativos a manejar una sola estructura global que sea compatible con las convenciones de internet.

A continuación se describen los aspectos relevantes de *Active Directory*:

Directorio	Es una estructura jerárquica de información acerca de objetos en la red.
Consultas	El directorio activo genera un catálogo global que pueden usar los usuarios y administradores para encontrar cualquier objeto en la red.
Administración	Maneja una lista mejorada de control (ACL) para controlar que los usuarios puedan ver y acceder a los objetos en el directorio activo.
Seguridad	Proporciona los mecanismos más fuertes y efectivos de seguridad con entidades externas como internet a través de una gran variedad de protocolos como son: Kerberos, protocolos basados en llave pública y autenticación de contraseña distribuida.
Duplicación	Dentro de cada dominio se duplica el directorio en cada servidor que esté ejecutando el directorio activo.
Particiones	Con el directorio activo, el directorio de cada dominio guarda información sólo acerca de ese dominio en lugar de utilizar un directorio masivo.
Integración con DNS	Utiliza el sistema de nombres de dominio (DNS) en lugar de la dirección TCP/IP de las computadoras en red.

Entre los elementos que deberemos diseñar y tener en claro desde el principio, son las estructuras Lógica y Física de *Active Directory*.

Arquitectura de *Active Directory*

La estructura de los servicios de *Active Directory* está compuesta de los siguientes elementos:

1. **Modelo de datos**. El modelo de datos de *Active Directory* se deriva del modelo X.500. El directorio contiene los objetos que representan los componentes de la red y cada objeto se describe por sus atributos. La colección de objetos que se pueden almacenar en el directorio se define en el esquema.

2. **Esquema.** Se define un conjunto de instancias de clases de objetos que se almacenan en *Active Directory*. El esquema se puede actualizar de manera dinámica; es decir, una aplicación puede extender el esquema con nuevos atributos y clases para utilizarla de manera inmediata. Las actualizaciones se realizan a través de la creación o modificación de los objetos esquema almacenados en el directorio. Los objetos esquema están protegidos por las listas de control de acceso (*Access Control List*, ACL) para que sólo los usuarios autorizados puedan modificar el esquema.

3. **Modelo de seguridad.** El directorio activo forma parte de la base informática de confianza de Windows Server y participa de manera completa en la seguridad. Las listas de control de acceso (ACL) protegen todos los objetos en el almacén de *Active Directory*, así como las rutinas de validación del sistema operativo.

4. **Modelo de administración**. Sólo los usuarios autorizados pueden realizar las tareas de administración en los servicios de *Active Directory*. Un usuario es autorizado por una autoridad mayor para que pueda ejecutar un conjunto de acciones específicas en un conjunto específico de instancias de objetos en algún subárbol identificado en el directorio, de manera que se realiza un control granular sobre quién puede hacer qué y permite la delegación de autoridad sin asignar privilegios mayores.

Arquitectura del servicio de directorios

La arquitectura de los servicios de *Active Directory* incluye los siguientes componentes:

- Agente del sistema de directorios (DSA, *Directory System Agent*). Construye una jerarquía de las relaciones padre-hijo almacenadas en el directorio. Proporciona API (Interfaz de Programación de Aplicaciones) para llamadas de acceso al directorio.

- Capa de la base de datos. Proporciona una capa de abstracción entre las aplicaciones y la base de datos. Las llamadas desde las aplicaciones nunca se realizan directamente a la base de datos, sino que sólo pasan a través de esta capa.

- Motor de almacenamiento extensible (ESE, *Extensible Storage Engine*). Comunica directamente con los registros individuales en el almacén de datos del directorio basándose en el atributo RDN. El RDN (nombre completo relativo) es una nomenclatura que utiliza *Active Directory* para referirse a los objetos.

- Almacén de datos (archivo de la base de datos Ntds.dil). Este archivo se manipula solamente mediante el motor de base de datos ESE (*Extensible Storage Engine*, motor de almacenamiento extensible). Se puede gestionar el archivo utilizando la herramienta: `Ntdsutil` el cual se genera cuando se instala Windows Server.

Espacio de nombre de *Active Directory*

I. Estructura lógica.

Los servicios de *Active Directory* son un espacio de nombres; es decir, un área acotada en el que se puede resolver un nombre. La resolución de nombres es el proceso de traducir un nombre en algún objeto o información que el nombre representa. El espacio de nombre está basado en el Sistema de Nombres de Dominio (DNS), el cual permite la interoperabilidad con las tecnologías de internet. La estructura lógica de *Active Directory*, comprende los siguientes componentes:

- **Objetos.** Un objeto es un conjunto de atributos con nombres diferentes que representa un recurso en la red**.** Los atributos del objeto son características de los objetos en el directorio. Por ejemplo, los atributos de un usuario pueden incluir su nombre, apellido, departamento y dirección de correo electrónico.

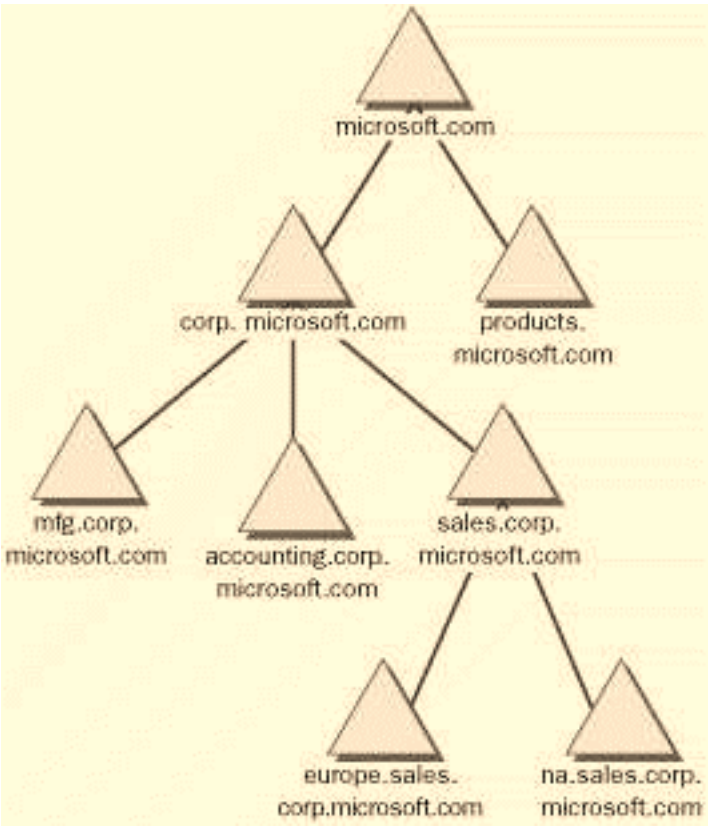
- **Dominios.** La unidad principal del directorio activo de Windows Server es el dominio, La agrupación de objetos en uno o más dominios, permite reflejar la organización de una empresa en la red, por ejemplo, **abc.org.** Todos los objetos de la red existen dentro de un dominio, y cada dominio almacena información únicamente de los objetos que contiene, el límite comprobado que puede contener es de un millón de objetos por dominio.
  Un dominio típico tiene los siguientes tipos de computadoras:

- **Controladores de dominio**. Computadora configurada como controlador de dominio que almacena y mantiene una copia del directorio.

- **Servidores miembros del dominio**. Computadora que no está configurada como un controlador de dominio, no almacenan información del directorio y

no puede autenticar usuarios. Los servidores miembros del dominio proporcionan recursos compartidos, como impresoras o carpetas compartidas.

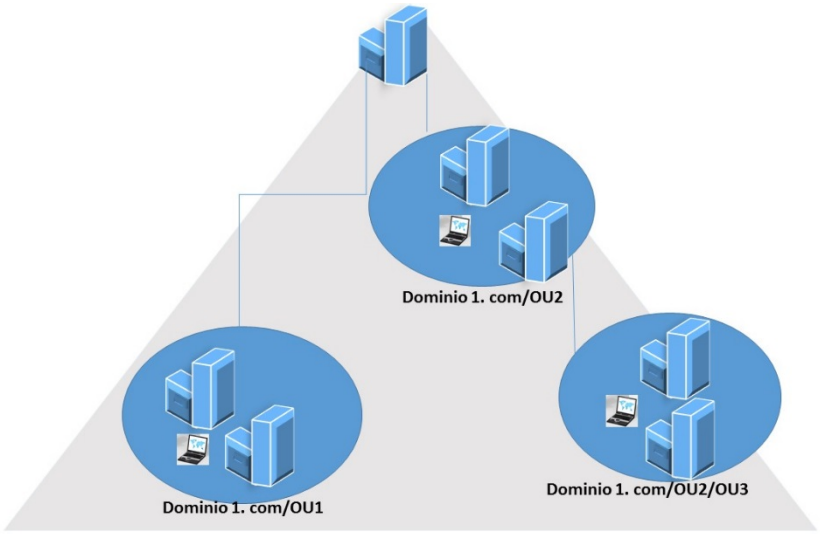
- **Computadoras cliente**. Computadoras con Windows que ejecutan un entorno de escritorio del usuario y permiten obtener acceso a recursos en el dominio.



(Imagen recuperada de http://support.microsoft.com/kb/310996/es, 23/01/2013)

- **Unidades organizacionales (OU).** Es un objeto contenedor lógico que se utiliza para organizar objetos dentro de un dominio. Se pueden colocar usuarios, grupos, PC y otras unidades organizacionales. La estructura organizacional está basada en departamentos o espacios geográficos de acuerdo a organigramas y responsabilidades administrativas. La jerarquía

de la OU dentro de un dominio es independiente de la estructura de otros dominios, por lo que cada dominio puede implementar su propia jerarquía OU. Como lo muestra la ilustración siguiente:

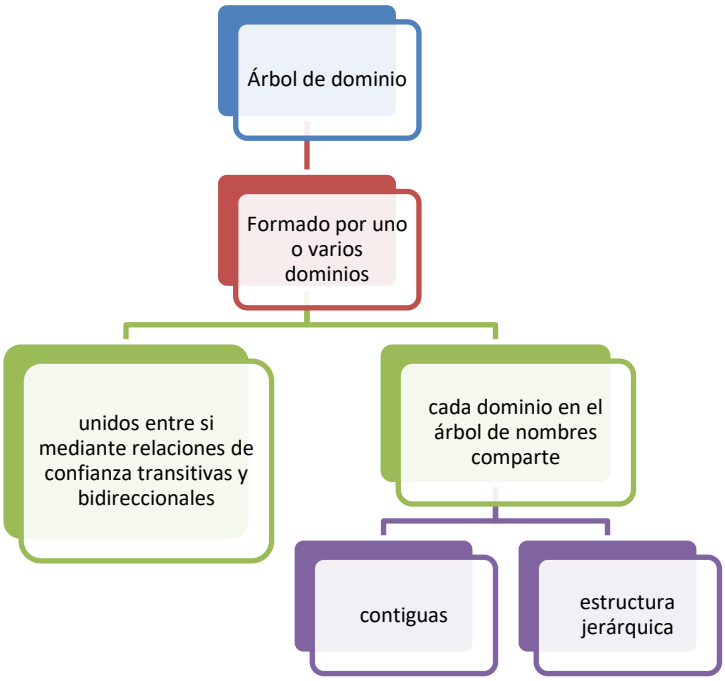


Elaboración con base en: http://bit.ly/1UmstNN

- **Árboles de dominio.** Un árbol es una agrupación u ordenación jerárquica de uno o más dominios; por ejemplo, Windows Server, que permite compartir los recursos de manera global. Un árbol puede consistir en único dominio Windows; sin embargo, se puede crear un espacio de nombre contiguo mayor uniendo múltiples dominios en una estructura jerárquica.

En *Active Directory* un árbol está definido por:

- Una jerarquía de dominios.
- Un espacio de nombres contiguo.
- Relaciones de confianza transitiva de Kerberos (Protocolo de autenticación de redes) entre los dominios.
- Un esquema común.
- Un catálogo global capaz de listar cualquier objeto en el árbol.

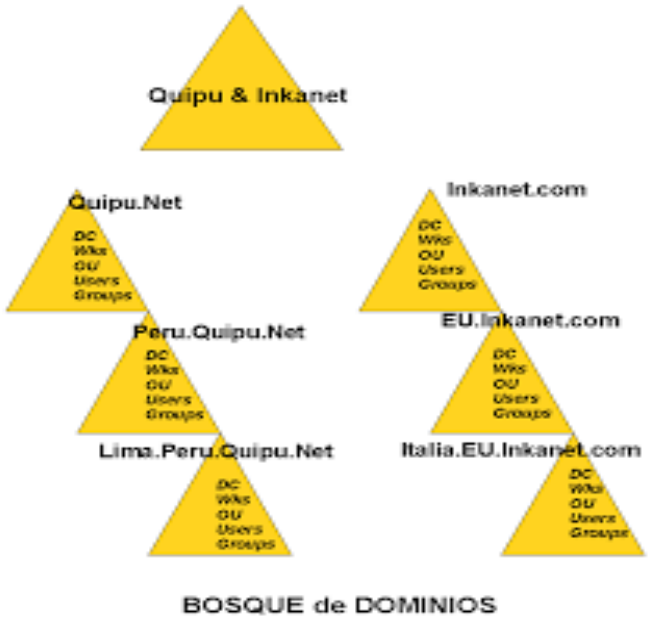


Elaborado con base en http://investigacionesir0109.blogspot.mx/2009/06/active-directory-en-windows-server-2003.html

- **Bosques.** Un bosque consiste en un conjunto de uno o más árboles que no forman un espacio de nombre contiguo. Los bosques permiten a las organizaciones agrupar divisiones (o que dos organizaciones combinen sus redes) que no utilizan el mismo esquema de denominación. Todos los árboles en un bosque comparten un esquema, configuración y catálogo global comunes.

Un bosque se define mediante:

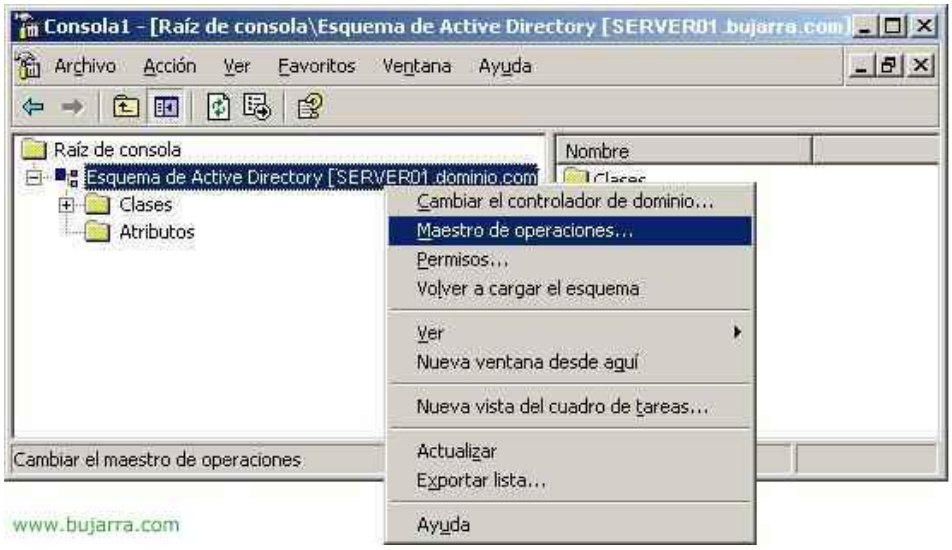
- Uno o más conjuntos de árboles.
- Espacios de nombres disjuntos entre estos árboles.
- Relaciones de confianza transitivas Kerberos entre los árboles.
- Un esquema común.
- Un catálogo global capaz de listar cualquier objeto en el bosque.



BOSQUE de DOMINIOS

(Imagen obtenida de http://jzel2222.blogspot.mx/2008/07/windowstechnologies.html, 23/01/2013)

- **Esquema.** El esquema del servicio de directorios, contiene las definiciones de todos los objetos, tales como computadoras, usuarios e impresoras que están almacenados en el directorio activo. La siguiente pantalla muestra la configuración del esquema de un equipo denominado: SERVER01.dominio.com.

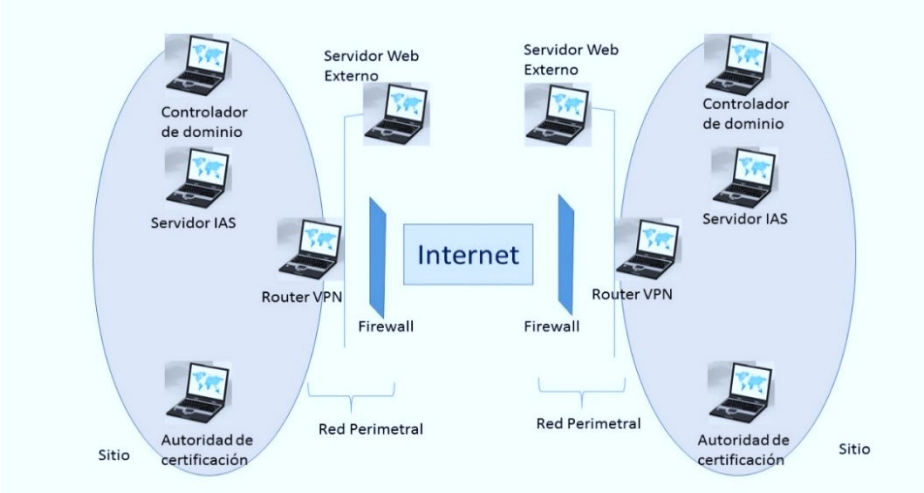


(Imagen obtenida de http://www.bujarra.com/ProcedimientoMigrarRoles.html, 23/01/2013)

II. Estructura física

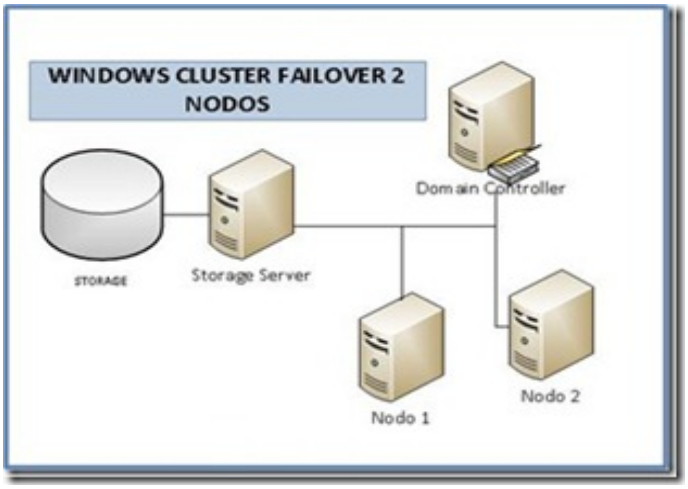
La estructura física de *Active Directory* optimiza el tráfico de datos determinando el lugar y el momento en que se produce la replicación de la información y el tráfico de la conexión. Para optimizar el uso del ancho de banda de red de *Active Directory*, se debe comprender la estructura física, cuyos elementos son los siguientes:

- **Sites.** Un *site* es una combinación de una o más subredes IP que están conectadas por medio de enlaces de alta velocidad, la razón principal para crear los *sites* es lograr la optimización del tráfico en la replicación y la conexión de los usuarios de manera rápida y confiable. Por ejemplo, un sitio puede definirse con los rangos de subred de 192.168.1.0/24 a 192.168.10.0/24. Otro sitio, en otro lado de un enlace remoto WAN, puede tener 172.20.10.0/24 a 172.20.20.0/24. Sin embargo los dos sitios pueden ser parte del mismo dominio de Windows Server. Un ejemplo puede verse en el siguiente esquema:



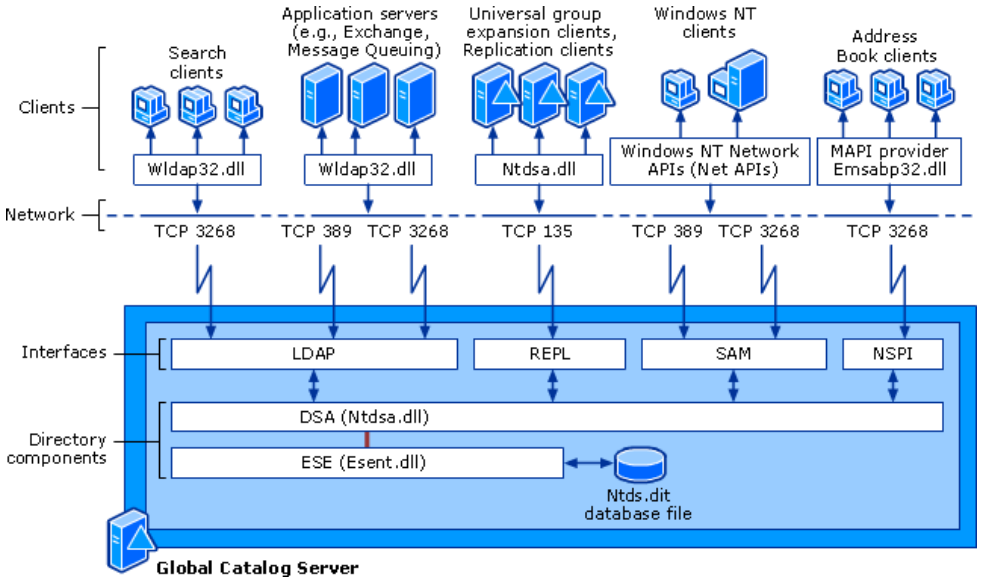
Elaboración con base en: https://technet.microsoft.com/en-us/library/cc775818%28v=ws.10%29.aspx

- **Controlador de dominio.** Es una computadora que está ejecutando el sistema operativo Windows Server y que almacena una réplica del directorio (Base de datos de dominio local). Todos los controladores en un dominio tienen una réplica completa de su porción del dominio del directorio. Cuando se actualiza el directorio, Windows Server automáticamente replica la actualización al resto de controladores de dominio. Por ejemplo, cuando existe un cambio de *password*, o cuando se bloquea una cuenta, se replican inmediatamente por todo el dominio.



(Imagen obtenida de http://blogs.itsynergy.co/agarciaf/2010/09/10/cluster-failover-dos-nodos-con-windows-server-2008/, 23/01/2013)

- **Catálogo global.** Es un servicio de almacenaje que contiene información de directorio de todos los dominios de una empresa. Su objetivo es responder consultas acerca de objetos en cualquier parte de la empresa a través de árboles de dominio.



(Imagen obtenida de http://technet.microsoft.com/pt-br/library/how-global-catalog-servers-work(v=ws.10).aspx, 23/01/2013)

- ***Single Master Operation*.** Un operador maestro es un equipo controlador de dominio que ha sido asignado para cumplir uno o más roles en el directorio activo de un dominio o bosque. En cada bosque existen al menos cinco funciones de maestro de operaciones que se asignan a uno o varios controladores de dominio. Las funciones de maestro de operaciones de todo el dominio deben aparecer una única vez en cada dominio del bosque.

Las funciones de maestro en todo el bosque son:

- Maestro de esquema. Este controlador de dominio controla todas las actualizaciones y los cambios que tienen lugar en el esquema de un bosque, debe tener acceso al maestro de esquema y sólo puede haber uno en todo el bosque.

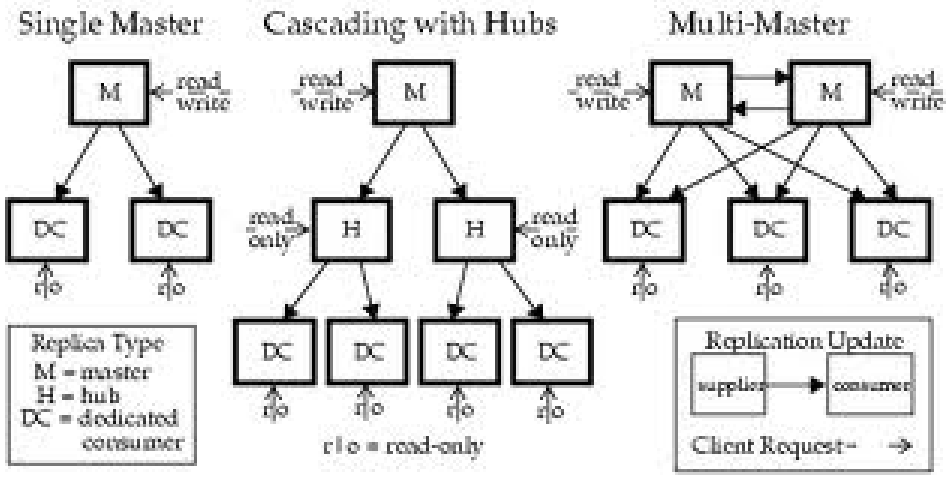
- Maestro de nombres de dominio. Este controlador de dominio controla la adición o eliminación de los dominios del bosque, sólo puede existir un maestro de nombres de dominio en todo el bosque.

Funciones de maestro de operaciones en el dominio:

Cada domino del bosque debe de tener las siguientes funciones:

- **Maestro de RID.** El maestro de RID asigna secuencias de *Id* (RID) a cada uno de los distintos controladores de dominio. Sólo puede existir un controlador de dominio con la función de maestro RID en cada dominio del bosque. Siempre que un controlador de dominio crea un usuario, grupo o un objeto, asigna un *Id* de seguridad (SID) único al objeto creado. Este SID se compone de un SID de dominio, que es el mismo para todos los SID creados en el dominio, y de un RID, que es único para cada uno de los SID creados en el dominio.
- **Maestro emulador PDC.** Si el dominio funciona con equipos cliente que no tienen el sistema operativo Windows, o si tiene controladores de dominio de reserva (BCD) de Windows NT, el maestro emulador de PDC actúa como controlador principal de dominio de Windows NT. El PCD procesa los cambios de contraseña de los clientes y replica las actualizaciones en los BCD. Sólo puede existir un controlador de dominio que actúe como maestro emulador de PDC en cada dominio del bosque. Este controlador admite los protocolos de autenticación Kerberos V5 y NTLM.
- Maestro de infraestructuras. Este controlador es responsable de actualizar las referencias de los objetos de su dominio en los objetos de otros dominios. El maestro de infraestructuras, compara sus datos con el catálogo global. Los catálogos globales reciben actualizaciones constantes de los objetos de todos los dominios a través de la replicación; de tal manera, que los datos de los catálogos globales siempre están actualizados. Si el maestro de

infraestructuras encuentra datos sin actualizar, solicita los datos actualizados a un catálogo global, posteriormente replica los datos actualizados en los demás controladores de dominio. Sólo puede existir un controlador de infraestructuras en cada dominio.



Escenarios de replicación

(Imagen obtenida de http://docs.oracle.com/cd/E19199-01/816-6698-10/replicat.html, 24/01/2013)

Beneficios de *Active Directory*:

Los beneficios de instalar *Active Directory* con Windows Server son principalmente:

- Se puede utilizar en una organización de cualquier tamaño: pequeñas, medianas y grandes.
- Sus servicios son seguros, distribuidos, con particiones y réplicas.
- Utiliza diversos protocolos (LDAP, Kerberos).
- Utiliza servicios de red como (DNS, DHCP, etcétera).
- Funciona como una base de datos que almacena de forma centralizada toda la información de un dominio con la ventaja de mantener la información sincronizada entre distintos servidores de autenticación de acceso a la red.

- Permite crear de forma centralizada una serie de objetos que están agrupados en categorías lógicas de una red; usuarios, grupos, recursos, permisos, políticas.
- *Active Directory* se replica de forma automática en todos los servidores que controlan el acceso al dominio.
- Permite crear estructuras jerárquicas de dominios y subdominios, lo que facilita la estructuración de los recursos según su localización o función dentro de la organización.

2.4. Sistema de archivos

De acuerdo con Robert William (William, 2004: 408-426), el sistema de archivos, ayuda a organizar de forma lógica los datos en los discos duros y proporciona al sistema operativo las rutinas necesarias para que puedan ser accedidos, modificados y eliminados. Los sistemas operativos suelen soportar varios sistemas de archivos; aunque sólo sea en modo lectura. Los sistemas de archivos de la familia Microsoft que podemos encontrarnos son: FAT (*File Allocation Table*) y NTFS (*New Technology File System*).

El sistema de archivos de Windows NT (NTFS) está diseñado para ofrecer una amplia seguridad dentro del sistema operativo Windows Server. La seguridad se logra a través de permisos que están disponibles en el volumen formateado con el sistema NTFS, a través de la combinación de carpetas compartidas y permisos, se provee a los usuarios el acceso y control a los recursos de la red. Los permisos NTFS no están disponibles en archivos con formato FAT para ediciones de Windows no empresariales (como lo es el caso de Server), ya que FAT no es redundante ante fallas y carece de permisos de seguridad en la que cualquier usuario puede acceder a cualquier archivo.

Aspectos importantes de NTFS

NTFS fue desarrollado para Windows NT para proporcionar un sistema integral de seguridad a los recursos; directorios, archivos, grupos, usuarios, aplicaciones, etcétera Con Windows Server tuvo mejoras adicionales como la capacidad de crear particiones dinámicas y cuotas (espacio) en los discos, cifrado (EFS) y desfragmentación, entre otros.

Para la protección de datos, el administrador de Windows Server debe saber cómo administrar los permisos con NTFS. Los permisos NTFS se dividen en dos categorías: los permisos de directorio y los permisos de archivo.

1- **Permisos de directorio.** Estos permisos controlan los archivos y subcarpetas que pueda visualizar un usuario y restringen lo que los usuarios puedan crear, eliminar, así como los cambios de permisos de las carpetas.

En la siguiente tabla se muestran los permisos de directorio:

Tipo de permiso	Acción de usuario permitida
Deny	El usuario no tiene acceso al directorio.
Read	El usuario puede ver el directorio y los archivos del directorio, puede ver atributos, propietarios y permisos.
Write	El usuario puede crear nuevos archivos y subcarpetas, cambiar los atributos de la carpeta y ver los permisos y propietarios.
List Folder Contents	Ver los nombres de archivos y subcarpetas en el directorio.
Read & Execute	Ejecutar acciones permitidas por los permisos de *Read* y los *Folder Contents*.
Modify	El usuario tiene los permisos de *Write y Read* & *Execute,* además de eliminar archivos.
Full Control	Cambiar permisos, tomar propiedad, borrar subfolders y archivos, además de ejecutar todas las acciones permitidas por los permisos anteriores.

2- **Permisos de archivo**. Como propietario, el usuario tiene el control total sobre los objetos y puede asignar permisos a otros usuarios para que accedan a ellos. Este control puede ser desde una denegación total del acceso, hasta el otorgamiento de propiedad de un archivo o carpeta a otro usuario.

En la siguiente tabla se muestran los permisos de archivo:

Tipo de permiso	Acción de usuario permitida
Deny	El usuario no tiene acceso.
Read	Leer el archivo y ver los atributos, propietarios y permisos.
Write	Sobrescribir en el archivo, cambiar los atributos y ver el propietario y los servicios.
Read & Execute	Ejecutar aplicaciones y acciones permitidas por el permiso de *read*.
Modify	Modificar y borrar archivos, además de las acciones permitidas por los permisos de *Write* y *Read* & *Execute*.
Full Control	Cambiar permisos, tomar posesión y ejecutar las acciones permitidas por los permisos anteriores.

Listas de control de acceso (ACL)

Las ACL son listas de control que tienen todos los objetos del directorio y permiten establecer permisos de diferentes propiedades a los objetos permitiendo o negando el acceso a la modificación de éstos a los diferentes usuarios o procesos del sistema. Su función es muy importante, ya que mediante las ACL, se pueden impedir accesos a carpetas y directorios y se puede delegar a ciertos usuarios ciertas tareas de una manera segura.

RESUMEN DE LA UNIDAD

Windows Server es un sistema operativo de la familia Windows Microsoft para funciones de servidor de archivos, impresión, aplicaciones y servicios Web. Este SO cuenta con un conjunto completo de servicios de infraestructura basados en *Active Directory*, cuyos servicios están orientados a centralizar la gestión de usuarios, grupos, servicios de seguridad y recursos de la red. El diseño que maneja es modular, a base de pequeños componentes de *software* independientes que trabajan juntos para realizar las tareas de un sistema operativo, y donde cada uno proporciona un conjunto de funciones que actúan como una interfaz para el resto del sistema.

Clave de este sistema operativo es, por una parte, el servicio de *Active Directory*, que provee la estructura y funciones para la organización, administración y control de los recursos de la red; y por otro, el sistema de archivos que maneja sobre Windows NT (NTFS), diseñado para ofrecer una amplia seguridad dentro del sistema operativo Windows Server. La seguridad se obtiene a través de permisos que se encuentran disponibles en el volumen formateado con el sistema NTFS.
Windows Server en conjunto con *Active Directory* se puede instalar para cumplir los requerimientos de empresas, pequeñas, medianas y grandes que deseen implementar el concepto de internet en el manejo de su información de manera segura.

BIBLIOGRAFÍA

SUGERIDA

Autor	Capítulo	Páginas
Microsoft Corporation	4	152-167 168-183
Microsoft Corporation	6	208-223 250-265

UNIDAD 3

GNU/LINUX

SUAyED UNAM
Sistema Universidad Abierta y Educación a Distancia

Plan 2012 2016 actualizado

OBJETIVO PARTICULAR

El alumno conocerá las características principales, operación y capacidad del sistema operativo GNU/LINUX.

TEMARIO DETALLADO

(8 horas)

3. GNU/LINUX

- 3.1. Características generales
- 3.2. Administración del sistema
- 3.3. Sistema de archivos

INTRODUCCIÓN

El *software* libre entró en el ámbito tecnológico con una fuerza tan grande que ya forma parte de los estándares a nivel mundial, su desarrollo ha sido impulsado por las comunidades académicas a través de internet. Un ejemplo de este tipo de *software* es el sistema operativo denominado GNU/Linux que está regido por lo que se conoce como la Licencia Pública General de GNU[10] o GPL (*General Public Licence*), lo que significa que el código fuente está siempre accesible. GNU/Linux se emplea para referirse a una combinación del *kernel* (núcleo) de uso libre denominado Linux, que se utiliza con herramientas del proyecto GNU. La GPL fue desarrollada para el proyecto GNU por la *Free Software Foundation* (Fundación por el *Software* Gratuito).

Es muy importante destacar que el **"***software* libre**"** de la GPL no es *software* de dominio público. El *software* de dominio público carece de copyright y pertenece literalmente al público. El *software* regido por la GPL, que no es gratuito, sí tiene copyright de su autor o autores. Esto significa que está protegido por las leyes internacionales del *copyright* y que el autor del *software* está declarado legalmente.

No sólo porque un programa sea de libre distribución puede considerársele de dominio público. El *software* regido por la GPL tampoco es "*shareware*". Por lo general, el "*shareware*" es propiedad del autor, y exige a los usuarios que le paguen cierta cantidad por utilizarlo después de la distribución. Sin embargo, el *software* que se rige por la GPL puede ser distribuido y usado sin pagar a nadie.

La GPL posibilita que los usuarios modifiquen el *software* y lo distribuyan; sin embargo, cualquier trabajo derivado de un programa GPL se regirá también por la

[10] http://www.gnu.org/licenses/licenses.es.html

GPL. El *software* libre se puede vender si se desea, aunque para ingresar en el mercado es necesario acompañarlo de servicios al usuario como: las actualizaciones, soporte técnico, distribución, mejoras, etcétera, es por esta causa que aparecen diferentes distribuciones de Linux como Red HAT, *Suse*, *Debian* y varias más, las cuales incluyen el beneficio de llevar a las manos de los usuarios las primeras versiones para su uso. En la presente unidad se abordan los aspectos más relevantes del sistema operativo GNU/Linux que, sin duda, ayudarán a decidir en qué proyectos puede utilizarse.

3.1. Características generales

GNU/Linux es un sistema operativo de libre distribución desarrollado originalmente por Linus Torvalds, estudiante de la Universidad Tecnológica de Helsinki. En un principio surgió como una idea para desarrollar un sistema operativo basado en MINIX[11] y siguiendo al estándar POSIX (*Portable Operating System Interface*) para mejorarlo. Actualmente ha tenido un fuerte desarrollo a través de miles de voluntarios de todo el mundo, que han contribuido a mejorar y añadir nuevas características al sistema. Como Linux es un proyecto de desarrollo abierto, todas las nuevas versiones que vayan apareciendo estarán disponibles al público, sean o no estables, para reconocer si una versión es o no estable, se ha acordado lo siguiente: las versiones 1.x.y, en la que cuando "x" es par, significa que es estable, cuando existe un incremento de "y", implica la corrección de un error. Por lo tanto, las versiones 1.2.2 a la 1.2.3 sólo hay corrección de errores. Las versiones 1.x con "x" impar son betas para los desarrolladores.

Distribuciones

Uno de los primeros conceptos que aparecen al iniciarse en Linux es el de la distribución, que consiste en un agrupamiento del núcleo del sistema operativo Linux y otra serie de aplicaciones de uso general, las distribuciones más conocidas son: Fedora Core (antes Red Hat), Debian, Mandrake, Slackware, SuSe y Corel Linux, todas ellas incluyen el *software* más reciente.

[11] Pequeño sistema operativo desarrollado por Andrew Tanenbaum con el fin de simular el funcionamiento de un sistema UNIX en una PC era utilizado por los estudiantes junto con su código fuente.

A continuación se describen las distribuciones más importantes:

- UBUNTU

Distribución basada en Debian, centrada en usuarios con pocos conocimientos técnicos, tiene una gran facilidad de uso, muy popular y con mucho soporte de la comunidad, el entorno del escritorio por *default* es GNOME.
Página Web: http://www.ubuntu.com/

- RED HAT ENTERPRISE

Distribución con muy buena calidad, contenidos y soporte a los usuarios por parte de la empresa que lo distribuye. Es necesario el pago de una licencia de soporte. Es un sistema operativo enfocado a empresas.
Página Web: http://www.redhat.com/

- FEDORA

Es una distribución de propósito general (*software* libre y código abierto), derivada de la distribución original de Red Hat Linux y soportada por la comunidad del proyecto Fedora. Es fácil de instalar, de buena calidad y muy estable, actualmente es una marca registrada de *Red Hat* que busca ser líder en la industria. Una de las ventajas más importantes es que se realizan los cambios en las fuentes originales en lugar de la aplicación de los parches tradicionales. Su desarrollo ha evolucionado de tal forma que aparecen nuevas versiones cada seis meses aproximadamente.
Página Web: http://fedoraproject.org/

- DEBIAN

Distribución de muy buena calidad, su proceso de instalación es más complicada, sin causar problemas. Sistema operativo de gran estabilidad operativa. Página Web: http://www.debian.org/

- Open SuSE

Distribución muy conocida a nivel mundial, fácil de instalar, administrar y utilizar, cuenta con una gran cantidad de asistentes gráficos para su instalación y configuración. Está orientada a usuarios comunes y desarrolladores experimentados, puede instalarse en estaciones de trabajo o servidores. Se puede descargar el *software* sin costo comercial. Página Web: https://es.opensuse.org/Bienvenidos_a_openSUSE.org

- SuSE LINUX ENTERPRISE

Sistema de muy buena calidad, contenidos y soporte a los usuarios por parte de la empresa que lo distribuye ("Novell"). Se requiere el pago de una licencia de soporte ya que está enfocado a servidores de alto desempeño. Página Web: https://www.suse.com/

- GENTOO

Es una distribución de GNU/Linux para usuarios que tienen experiencia en este sistema operativo. Está basado en el sistema de distribución de *software*: Portage BSD-port. *Portage* contiene las descripciones de los paquetes de *software* y *scripts* necesarios para la instalación del sistema a través de la sincronización con un

servidor remoto. Para su instalación se requiere buena configuración de *hardware* para el servidor, acceso a internet de buena calidad (ancho de banda) ya que las versiones de *software* se actualizan de manera continua y pueden requerir mucho tiempo si el *hardware* y la red están limitados.
Página Web: http://www.gentoo.org/

- KUBUNTU

Es una distribución basada en Ubuntu de código abierto y construido en base al *kernel* (núcleo) de Linux. Está centrada en el usuario final y facilidad de uso. La diferencia importante con Ubuntu es que el entorno del escritorio por *default* es KDE (K *Desktop Environment*) que es una plataforma de desarrollo y solución de escritorios líder en sistemas Unix/Linux. *Kubuntu* puede utilizarse en entornos de escritorio y servidor para usos domésticos o comerciales.
Página Web: http://www.kubuntu.org/

- MANDRIVA

Distribución creada en 1998 con el objetivo de acercar a los usuarios al uso de Linux y al *software* libre, en un principio se llamó *Mandrake Linux*. Es de mucha facilidad de uso para todo tipo de usuarios. Sus principales características son: disponible en 74 idiomas, instalación muy amigable a diferencia de otras distribuciones, utiliza *Mandrake Control Center* para su administración, proporciona entornos de escritorio KDE y Gnome, puede instalarse en múltiples plataformas de *hardware* para estaciones o servidores.
Página Web: http://iso.linuxquestions.org/mandriva/

- SLACKWARE

Es una de las primeras distribuciones que existió y fue la primera en alcanzar un uso masivo, está orientado para las personas que desean aprender, por lo que está basado en menús y se ajusta a lo que realmente se quiere realizar. Es una distribución muy estable para estaciones y servidores. Página Web: http://www.slackware.com/

Características

Todas las versiones incluyen lo siguiente:

- Entorno gráfico.
Linux puede funcionar en entorno gráfico o en modo consola. El entorno gráfico está orientado al usuario final, mientras que el modo consola es más usado en distribuciones para servidores, por ejemplo: servidores de bases de datos, aplicaciones Web, administración de redes; etcétera

- Sistema de programación.
GNU tiene capacidad para compilar en los lenguajes: C, CC+, Java, etcétera, también puede soportar diversas arquitecturas mediante la compilación cruzada (compilación de código fuente para una arquitectura diferente) ideal para desarrollos heterogéneos. Existen diversos entornos de desarrollo integrado para GNU/Linux que incluyen: *Kdevelop, Ultimate*++, Adjunta, Eclipse e IDE. GNU tiene capacidad para programar a través de Shell (intérprete de comandos), procesador de textos por patrones, y la mayoría de las distribuciones cuentan con los lenguajes de programación: Python, Perl, PHP y Ruby.

- Aplicaciones de usuario.

Las aplicaciones para los usuarios se distribuyen con los paquetes de archivos en los formatos **.deb** y **.rpm**, también se pueden instalar aplicaciones a través del código fuente de todas las distribuciones.

- *Software* de código cerrado para GNU/Linux

A través del tiempo, se han ido incorporando aplicaciones de código cerrado para GNU/Linux, entre los que destacan: Adobe Reader, Adobe Flash, Opera, Google Picasa, etcétera.

Arquitectura

El término “arquitectura” se refiere a la forma de cómo se organizan los diferentes elementos del sistema operativo para que adquiera una identidad y un modelo de funcionamiento acorde con la Base Estándar para Linux (LSB). La LSB es un proyecto del *Free Standars Group* que tiene como objetivo crear y normalizar la estructura interna de los sistemas operativos derivados de Linux. LSB especifica: sistema de archivos, bibliotecas estándar, niveles de ejecución, sistemas gráficos, formatos de distribución, etcétera.

En la siguiente figura se muestran los elementos de la arquitectura Linux:

- Como primer elemento está el núcleo o *kernel* el cual conecta el exterior con el *hardware*.
- Shell que se encarga de manejar todos los comandos, interpretarlos y ejecutarlos.
- Alrededor del *shell* están los programas y comandos.
- Por último están las aplicaciones desarrolladas para tareas específicas y que se instalan en la capa de aplicaciones.

- Manejo de tareas

Linux maneja multitareas, esto se refiere a que múltiples tareas trabajen al mismo tiempo. También es multiusuario, esto consiste en que múltiples usuarios se conecten al sistema al mismo tiempo para la utilización de los recursos (*hardware*, CPU, memoria, etc.).

- Manejo la Unidad de Proceso Central (CPU)

Administra el recurso más importante, el CPU, a través del proceso de interrupciones realizado por el núcleo.

- Administración de tareas

Asigna las tareas al conjunto de recursos durante lapsos para luego interrumpirlas y asignarlo a otra tarea. Tiene además, el mecanismo de prioridades que permite al administrador del sistema, otorgar prioridades diferentes, de acuerdo a la importancia de cada tarea.

- Manejo de interrupciones

Por medio del manejo de interrupciones el sistema operativo obedece las señales que el *hardware* le hace sobre el inicio o terminación de asignaciones de trabajo y con el cual se orienta para administrar mejor las tareas. El *hardware* se comunica a través de los requerimientos de interrupciones (IRQ) que son números asignados a los dispositivos para identificar sus señales al núcleo.

- Interprocesos

Los interprocesos son tareas que se ejecutan y que en un momento dado pueden comunicarse entre sí en un programa; es decir, existe comunicación entre procesos (IPC), éstos deben estar sincronizados para evitar caídas del sistema. Este mecanismo tuvo su origen en el sistema operativo Unix, en el cual un proceso, que produce una salida, la envía a una salida estándar que puede ser la pantalla, archivo o bien como entrada a otro proceso a través del símbolo pipe “|”. Por

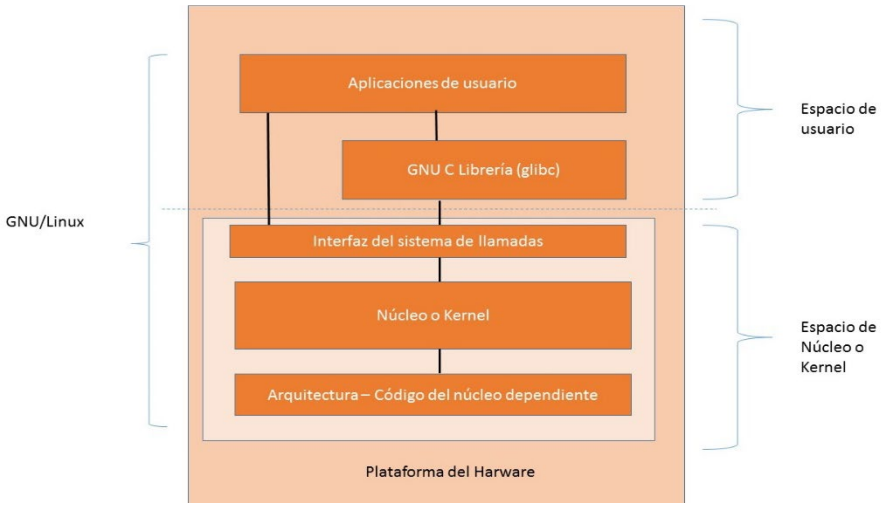
ejemplo, listar los archivos de un directorio y guardar los resultados en un archivo llamado “prueba”. La instrucción es: ls –l > prueba.

- Memoria

El sistema operativo administra el uso de la memoria a través del mecanismo de jerarquías, con la asignación de la memoria a las tareas que la necesitan y la proporción de seguridad para que la memoria reservada por una tarea no sea usada por otra.

- Interfaz del *hardware*

Las interfaces comienzan con las aplicaciones que ve el usuario en su pantalla, manejadas por los programas de aplicación, las cuales utilizan las API (Interfaces de Programas de Aplicaciones), que son las que permiten conectar con los programas de utilidad y el núcleo del sistema. Por ejemplo, cuando un usuario lee un documento, esta acción se traduce en el interior del sistema en una orden a un *driver*, que es un programa que interactúa con el disco duro para leer uno o varios datos en cadena que se encuentran en los dispositivos del *hardware* (CPU, Bus, video, teclado, etcétera). La siguiente figura muestra la interfaz del *hardware* (*Hardware Plataform*) y su interrelación con las aplicaciones del usuario (*User Aplications*) en la arquitectura de GNU/Linux.



Elaboración con base en: http://bit.ly/1WYiCyq

- Dispositivos e interfaces principales.
  - CPU. GNU/Linux soporta arquitecturas de 32 y 64 *bit*s.
  - Buses. Soporta los estándares; IEEE-696, ISA, EISA, PCI y Multibuses.

- Entrada y salida en paralelo. Permite enviar *bytes* de manera simultánea, por lo que es más rápido. Por ejemplo, una impresora que se comunica a través de buses paralelos.

- Entrada y salida en serie. Permite la comunicación lenta y asíncrona a través de la interfaz RS-232.

- Memoria de acceso directo (DMA). Permite que los dispositivos de diferentes velocidades se comuniquen sin realizar interrupciones masivas para que el CPU siga siendo útil a otras tareas.

3.2. Administración del sistema

En esta sección se abordará el tema de la administración de este sistema operativo, estando en el entendido de que ya se encuentra instalado en el equipo[12].

Cuando ya se ha instalado alguna distribución de GNU/Linux, viene un proceso importante en el aprendizaje sobre su uso. Este aprendizaje debe desarrollarse de manera práctica y paulatina conociendo el funcionamiento de diversos comandos y herramientas que permitirán desarrollar al alumno un criterio tecnológico necesario para tomar decisiones relacionadas con la adopción de este sistema como plataforma para ofrecer soluciones a las organizaciones. Uno de los primeros acercamientos es el poder acceder al sistema, lo cual se puede realizar a través de algún tipo de cuenta y contraseña. Las cuentas pueden ser de tipo; superusuario (*root*), cuentas especiales para la administración del sistema ("admin", "sa", etcétera) y cuentas comunes de usuario, que de acuerdo a sus capacidades, harán uso de los recursos disponibles en el mismo. La cuenta de superusuario (*root*) tiene privilegios absolutos sobre todo el sistema operativo, y esto incluye su configuración, comandos empleados, protección, mantenimiento, herramientas, gestión de usuarios y grupos, contraseñas, etcétera. Es muy importante mencionar que la pérdida del *password* de la cuenta "*root*" puede comprometer seriamente la seguridad del equipo y sus recursos.

Las principales acciones que están relacionadas con la administración del sistema GNU/Linux y que estaremos revisando en este tema son:

[12] En Internet puedes encontrar vasto material acerca de la instalación del sistema operativo de acuerdo a la arquitectura de *hardware* que tengas; es recomendable que hagas esta investigación si es tu primer acercamiento con este *sistema operativo*.

1. Acceso al sistema

Para acceder y navegar en este sistema, se requiere del entorno gráfico o, bien, del intérprete de comandos "*Shell*". Los entornos gráficos para GNU/Linux han evolucionado mucho en los últimos años. Se puede utilizar el sistema X Windows y prácticamente se puede empezar a utilizar el sistema; sin embargo, es recomendable aprender a trabajar desde el intérprete de comandos "*Shell*"; además de que es más rápido que el entorno gráfico.

En la mayoría de las distribuciones Linux, cuando el sistema se carga en memoria, se iniciará por *default* el sistema gráfico: Gnome[13], KDE[14] u otro. Cuando no se ha instalado ninguno de los escritorios, se presentará en pantalla, en la parte inferior, lo siguiente: ***Bastion login*:**

Bastion hace referencia al nombre que se le haya asignado al servidor durante el proceso de instalación, a continuación se deberá teclear el usuario y la contraseña solicitada. En cualquier caso siempre será necesario contar con una cuenta de acceso y su contraseña. Se puede acceder desde el entorno gráfico al intérprete de comando seleccionando las ventanas correspondientes. Por ejemplo, en la distribución *Red Hat* se selecciona: **Menú principal > Herramientas del sistema > Terminal.** En otra distribución el acceso es similar.

Cuando se accede al intérprete de comandos, aparece lo que se conoce como consola, que es una pantalla y un "*prompt*" visible. El *prompt* es un símbolo indicativo que espera a recibir órdenes, suele representarse por el nombre del equipo, separado por dos puntos y el nombre del usuario seguido del signo # o $ dependiendo del tipo de usuario. Si el usuario es *root* se representa con #, si es un usuario común el signo es $. Ejemplo. "Servidor1: carlos $" el equipo es Servidor1 y el usuario es Carlos. El formato del "*prompt*" dependerá de la distribución instalada

[13] Sitio oficial de GNOME: http://www.gnome.org/. Versión hispano: http://es.gnome.org/
[14] KDE: http://planetkde.org/es/

y puede ser configurado para mostrarse de otra forma. Al empezar a navegar en el sistema, es muy fácil perderse u olvidar en qué directorio del sistema nos encontramos, el comando "`pwd`" (*print working directory*) nos muestra la ubicación exacta del directorio actual. Po ejemplo, al teclear el comando `pwd` nos muestra `/home/sam` lo que indica que está en el directorio "sam", que a su vez está dentro de su escritorio principal /home.

Otro elemento importante es conocer el nombre del equipo. Cada sistema Linux tiene un nombre de sistema (*hostname*) asignado durante la instalación. Este "*hostname*" le da nombre a la máquina y es utilizado para identificar los equipos en una red, si durante la instalación se omite dar este nombre el sistema, asignará el nombre "*localhost*" o el nombre de la distribución que se instaló.

2. Administrador o *Root*

Como se mencionó al inicio de la unidad, el usuario administrador, *root* o superusuario, es una cuenta especial del sistema operativo que tiene acceso completo a todos los archivos y programas. Cualquier acción que ejecute *root* será obedecida sin solicitar confirmación, pudiéndose correr el riesgo de borrar o modificar archivos importantes que pueden dejar al sistema operativo inutilizable o muy difícil de reparar. Para evitar esto, se deben de generar cuentas de acceso al sistema con permisos limitados (diferentes a *root*); además, es recomendable tener más experiencia en el manejo del sistema operativo antes de intentar hacer una modificación que pueda afectarlo considerablemente.

→ Ejecutar acciones en el sistema como *root*

En GNU/Linux, la gran mayoría de los archivos importantes y programas para la configuración del sistema sólo pueden ser utilizados por el superusuario. Hay dos maneras de ejecutar estos programas: con el comando ***sudo*** y el comando ***su***. El uso de estos comandos permite limitar el uso excesivo de la cuenta de

superusuario, evitando así el riesgo de ocasionar algún daño al sistema operativo y sus aplicaciones.

→ SUDO

El comando **"sudo"** (*Superuser* DO) posibilita ejecutar un programa desde la línea de comandos con permisos de superusuario e implementa un control de acceso a usuarios normales que requieren ejecutar comandos con capacidades de *root*; su estructura es la siguiente:

sudo comando argumentos, por ejemplo: ***sudo –e /etc/inittab***

Por medio de esta acción es posible modificar el archivo "`inittab`" como si fuera *root*. Al ejecutar **sudo**, el sistema operativo solicitará la contraseña de *root*, la cual se tecleará para ejecutar comandos con privilegios de superusuario.

→ SU

Con el comando **"su"** se puede cambiar de usuario y acceder con otro usuario directamente desde la línea de comandos, el formato es el siguiente; **su nombredelusuario**, por ejemplo; **$ su root <enter>** el sistema solicitará la contraseña de superusuario.

3. Comandos básicos del sistema

Una vez que se está dentro del sistema, se emplean ciertos comandos para poder acceder a información, organizar archivos, etcétera. Es importante conocer la función de algunos comandos y familiarizarse con ellos.

La siguiente tabla muestra algunos comandos generales de la consola GNU/Linux:

Comando	Acción
man	Muestra un manual explicativo de los comandos, su organización y forma general de uso.
cd	Facilita el movimiento a través de los directorios del sistema.
ls	Enlista todos los archivos y carpetas dentro del directorio actual.
cp	Este comando realiza copias únicamente de archivos, mas no de directorios.
mkdir	Se utiliza para crear directorios.
mv	Es utilizado para mover o renombrar archivos.
rm y rmdir	rm borra archivos. rmdir borra directorios, siempre y cuando estén vacíos; de lo contrario se utilizará el comando rm –r.
whoami	Posibilita conocer con qué usuario estamos trabajando.
id	Se utiliza para conocer datos del usuario con el que se está trabajando y también los grupos al que pertenece.
uname	Es utilizado para conocer los datos generales del sistema operativo.
hostname	Muestra el nombre del equipo.
w	Visualiza a los usuarios conectados en la máquina.

Otros comandos que se emplean, son:

→ Uso de discos con “du” y “df”

Otros comandos que se emplean en este sistema, es el comando “*du*”, con él se conoce el tamaño en disco de los archivos y directorios, mientras que “*df*” indica el espacio libre y usado en las diferentes unidades del sistema; también se aplica para unidades extraíbles tales como discos externos, CD, etcétera.

→ Leer el correo: “mail”

Mail es un comando que permite consultar y enviar correos electrónicos que se encuentran en el buzón del usuario en modo texto. Para ejecutarlo se utiliza la palabra “mail”, ejemplo; **$mail [nombre-destinatario].**

4. Editor de textos en modo consola: “vim”

GNU/Linux incluye diferentes editores de texto que sirven para editar algún archivo del sistema o bien para generar algún documento, se recomienda el uso de “vim” ya que se puede encontrar tanto en Linux como en Unix, debido a que es un potente editor multifuncional de archivos, eficiente y posibilita el máximo rendimiento con el mínimo esfuerzo.

El modo de edición en “vim” funciona de dos modos:

a. El modo consola, que es cuando se inicia el programa, y

b. El modo de inserción, con el cual se puede modificar el archivo.

Una vez que se realizaron las modificaciones en el archivo, se presiona la tecla **ESC** para regresar al modo consola.

Para mayor información de este editor, puedes consultar el siguiente documento disponible en este sitio: **http://es.tldp.org/Tutoriales/doc-tutorial-vim/Guia-Vim.pdf**

La siguiente tabla muestra los comandos principales de edición de **“vim”**:

Comando	Acción
:q	Salir, cuando no se realizaron cambios.
:q!	Salir, sin guardar cambios.
:u	Deshacer el último cambio.
:w	Guardar los cambios.
:wq	Salir, guardando los cambios.
:x	Salir, guardando los cambios.
:edit	Abre un archivo para editarlo.
ESC	Regresa a modo comando.
i	Entrar en modo inserción.
v	Pasar a modo visual, es posible seleccionar texto.
/<texto>/	Buscar alguna palabra en el texto.

5. Instalación de programas

En la parte correspondiente a la instalación de programas en GNU/Linux, podemos encontrar dos paquetes[15] particulares para ello, uno se refiere a RPM (*Red Hat Package Manager*), y a la herramienta Yast.

Los RPM fueron desarrollados por *Red Hat* con la finalidad de instalar, actualizar y eliminar más sencillamente programas pre-compilados en Linux. Con esto se logró evitar el proceso anterior que requería compilar y generar los archivos adecuados para su correcto funcionamiento.

[15] O set de programas pre-armados para una función en particular.

Por otro lado, la empresa Novell desarrolló la herramienta **"YaST[16]"** y su más reciente mejora el *One Click Install* (instalación con un clic). Es una herramienta, actualmente de código abierto bajo la licencia de GPL de GNU, para la instalación y administración de programas, principalmente de la distribución de Linux OpenSUSE y SUSE.

Cuando se llama a "YaST" se muestra una pantalla similar a la generada por el entorno gráfico. Dentro de este entorno se puede desplazar entre las opciones utilizando el cursor.

También se pueden instalar paquetes sin utilizar el entorno gráfico de YaST, y esto se logra a través de la línea de comandos.

Las modificaciones, instalaciones, desinstalaciones, manejo de actualizaciones de los paquetes y demás se pueden manejar de modo centralizado utilizando esta herramienta.

La siguiente tabla muestra los comandos principales para la instalación:

Parámetro	Descripción
-List	Lista los paquetes instalados y sus versiones.
--install paquete	Instala el paquete seleccionado.
--update paquete	Actualiza el paquete seleccionado.
--remove paquete	Borra el paquete seleccionado.
--status	Muestra el estado de un paquete.

[16] Acrónimo de Yet another Setup Tool, cuya traducción aproximada es *"Otra Herramienta de Configuración Más"*

6. Registros del sistema

GNU/Linux fue diseñado para que al administrador del sistema le permita conocer la información que necesita sobre el equipo que se administra; esta información es suministrada por medio de "*logs*", que son archivos de texto ordenados de forma cronológica, comenzando por los registros más antiguos, de tal manera que se pueda conocer todo lo que pasa por el sistema; contempla aspectos de seguridad, registros del sistema, errores de instalación, accesos no autorizados, etcétera.

7. Niveles de ejecución del sistema

Lo primero que ejecuta el *kernel* es el programa "init" para gestionar el acceso a los recursos, define qué servicios están operando y el orden de los mismos, el cual se identifica mediante números. Este programa es el padre de todos los procesos, ya que se encarga de iniciar al resto de los *daemons,* "demonios" (programa que controla algún servicio).

En esta etapa es cuando entran en juego los niveles de ejecución (también conocidos como *runlevels* o niveles de corrida) de un equipo.

La siguiente tabla muestra los niveles de ejecución y la numeración que tiene asignado cada uno:

Runlevels	Descripción
0	Apagado.
1	Modo monousuario (sin gestión de red).
2	Multiusuario local (sin gestión de red).
3	Multiusuario completo con gestión de red.
4	No usado.
5	Runlevel 3 con gestor de ventanas (Servidor x).
6	Reinicio.

Los "*Runevels*" sirven para determinar en qué estado se requiere poner el servidor y son muy útiles en las pruebas que se realizan al sistema.

8. Planificación de tareas

La planificación de tareas sirve para poder ejecutar tareas en el sistema de manera automática a una determinada hora. El comando "*cron*" es utilizado para realizar esta función, comúnmente se activa por el administrador al inicio del sistema, y es éste el que ejecuta las tareas definidas en los archivos "*crontab*".

Crontab comprueba qué usuarios pueden o no utilizarlo y el formato que maneja es el siguiente:

[Minuto][Hora][Día del mes][Mes del Año][Día de la semana][Comando]

9. Análisis de procesos y rendimiento

Una de las principales funciones del administrador del sistema es comprobar periódicamente el estado que guarda el sistema operativo, así como su funcionalidad. Para lograr esto, se pueden ejecutar desde la línea de comando "Shell" diversos comandos que permiten su comprobación. La ejecución puede realizarse desde la cuenta de superusuario, la cual mostrará de manera completa todas las opciones disponibles. También pueden ejecutarse algunos comandos desde la cuenta de un usuario común; pero mostrará los resultados de manera más limitada. Los comandos más comunes son:

ps	Obtiene una vista de los procesos que se están ejecutando en el sistema.
top	Proporciona información en tiempo real sobre las tareas manejadas por el *kernel*.
vmstat	Proporciona información sobre procesos, memoria, paginación, bloqueo de entrada/salida, trazas y actividad del CPU.
free	Muestra la cantidad de memoria virtual que tiene libre el sistema.
iostat	Monitorea la carga de los diversos dispositivos de entrada/salida de un sistema (incluyendo los remotos) ponderándolos por sus tasas de transferencia.

10. Respaldos del sistema

Los sistemas GNU/Linux se encuentran expuestos a diversas interrupciones que pueden afectar su operación y dañar o perder información, las más comunes son; fallas de energía, errores de *hardware* y *software*, errores humanos, virus, caídas de red, hackers, inundaciones, fuego, etcétera. Aunque no se pueda prevenir cada una de estas interrupciones, el administrador del sistema sí puede prepararse para evitar las consecuencias si esto llega a ocurrir. Esto se logra por medio de la realización de respaldos (*backup*) y poder, de esta forma, restaurar la información dañada. Las técnicas empleadas para realizar respaldos han evolucionado en los últimos años debido principalmente a la capacidad de almacenamiento requerido y a la tecnología de redes de almacenamiento de tipo SAN (*Storage Área Network*). Los respaldos pueden realizarse a través de soluciones existentes en el mercado, o bien ejecutando diversos programas para GNU/Linux. También debe de considerarse el tipo de medio para realizar los respaldos; cintas, discos, etcétera. Se tendrá que realizar un estudio sobre las ventajas y/o desventajas para utilizar un medio en particular.

Existen cuatro tipos de *backup*:

1. Desestructurado	2. Completa + Incremental
No existe una metodología a seguir y se almacenan las copias de los archivos que se consideran más importantes.	Se realiza una copia completa del sistema y posteriormente se almacenan copias periódicas, con las diferencias que tienen con respecto al primer *backup*.

3. Espejo + Diferencial	4. Protección continúa de datos
A diferencia del segundo, en éste se almacenan las copias periódicas del sistema y al final se almacena una copia completa. De esta manera, es más eficiente recuperar el último estado estable, y no se tienen que rehacer los cambios desde el último *backup* completo.	Registra continuamente cada cambio que se realiza en el sistema original.

Los principales programas para realizar los respaldos son; ***dump, tar* y *cpio***, para restaurar la información se utiliza ***restore***.

→ dump y restore

El programa "*dump*" respalda un sistema de archivos completo en un dispositivo, lee directamente el sistema de archivos y no del sistema de archivos, fue desarrollado específicamente para hacer copias de seguridad. Leer el sistema de archivos de forma directa tiene algunas ventajas ya que puede realizar copias sin afectar las marcas de tiempo de los archivos. El programa "*dump*" de GNU/Linux puede leer únicamente el sistema de archivos ext2 (ver el tema de sistema de archivos) El programa "*restore*" obtiene de nuevo las copias y las restaura al sistema original. Es posible respaldar y restaurar datos a través de una red con una unidad de cinta conectada a otra computadora con los programas; ***rdump y rrestore***.

→ ***tar y cpio***

Los programas "*tar*" y "*cpio*" son empaquetadores de archivos similares, ambos son capaces de almacenar y recuperar archivos en cintas, aunque también son capaces de utilizar prácticamente cualquier medio, esto debido a que los controladores de dispositivos del *kernel* (núcleo) son los que se encargan del acceso al *hardware* a bajo nivel. Para respaldar con estas herramientas, es necesario que primero se monte el sistema de archivos con permisos de solo-lectura; y para utilizarlas es muy importante consultar el manual de la distribución GNU/Linux, por las diferentes opciones que ofrecen.

11. Usuarios y grupos

GNU/Linux es un sistema multiusuario en la que varias personas pueden trabajar de manera simultánea en el sistema protegiendo los datos de cada usuario, también es multitarea en la que varios procesos de varios usuarios pueden ejecutarse a la vez y de tiempo compartido en la que todos los procesos se ejecutan todo el tiempo. Los usuarios y grupos tienen las siguientes características:

- **Cuentas de usuario.** Contiene datos e información de cada usuario con un nombre y contraseña para acceder.
- **Grupos de usuarios.** Cada usuario puede pertenecer a uno o varios grupos.
- **Permisos.** Sistema de protección de datos de un usuario respecto de otros.

Para la administración del sistema existen tres cuentas importantes:

→ *Cuenta de usuario root*: Esta cuenta hace referencia al superusuario normalmente llamado *root*. Con esta cuenta se puede tener acceso a todos los archivos y se pueden ejecutar ciertos programas, por ejemplo, sólo *root* puede levantar servicios "demonio" (procesos que no son controlados por el

usuario). *Root* tiene un UID (*User Identification*) de 0, cualquier cuenta de usuario que cuente con un UID 0 tiene propiedades de *root,* por lo que es importante verificar si dicha cuenta justifica que tenga este tipo de privilegios por los daños que puede llegar a ocasionar al sistema.

→ *Cuenta de sistema*: Estas cuentas son usuarios del sistema operativo creadas durante su instalación, son utilizadas por procesos o tareas administrativas que se llevan a cabo en el mismo sistema. Estas cuentas no tienen un *password* o *shell* válidos, lo que impide que puedan ser utilizadas por usuarios normales, algunas cuentas son: lp (asociado al servicio de impresión), uucp (usuario asociado a UUCP), irc (usuario asociado a servidores de IRC), shutdown (asociado al apagado del sistema), no todas las cuentas se utilizan, depende de las funciones del sistema. Los rangos más altos de UID son reservados para cuentas especiales, por ejemplo *nobody* tiene un UID de 65534.

→ *Cuenta normal:* es una cuenta de usuario normal que se utiliza para validarse en el sistema. Los usuarios normales cuentan con privilegios reducidos para restringir el acceso a funciones sensibles del sistema. El UID utilizado para este tipo de cuentas oscila entre valores de 1000 en sistemas openSUSE y sobre 500 en sistemas Fedora.

Los usuarios normales cuentan con privilegios reducidos para restringir el acceso a funciones sensibles del sistema. Todas las cuentas creadas de usuarios se almacenan en el archivo /etc/passwd. Por ejemplo, si se ejecuta el comando ~>cat /etc/passwd se listan todos los usuarios creados en el sistema y tiene el siguiente formato:

pedro:x:1002:100:Pedro Fuentes:/home/pedro:/bin/bash

Cada campo está separado por dos puntos.

La siguiente tabla muestra lo que significa un formato de usuario:

pedro	El nombre de usuario, debe ser único en la máquina local.
x	Campo de contraseña, se aloja en el archivo /etc/shadow.
1002	Corresponde al identificador de usuario, comúnmente referido como UID (*User Identification*).
100	Este valor corresponde al identificador de grupo comúnmente referido como GID (*Group Identification*). Este identificador es único en el CPU y se ocupa de resguardar un rastro de cuáles archivos pertenecen a ese grupo en particular.
Pedro Fuentes	Este campo sirve para guardar los comentarios sobre el usuario, normalmente se llena con el nombre completo del usuario.
/home/pedro	Este campo hace referencia al directorio de inicio del usuario.
/bin/bash	Este campo contiene la consola por defecto que ocupa el usuario para ejecutar instrucciones en el equipo.

Comandos para la administración de usuarios y grupos:

Comando	Descripción
chague	Modifica o establece los atributos del *password*.
chfn	Modifica la función del *finger*, campo comentario.
chsh	Cambia el *Shell* del usuario.
finger	Despliega información sobre los usuarios del sistema.
gpasswd	Administra los usuarios en los grupos.
groupadd	Agrega un nuevo grupo al sistema.
groupdel	Elimina un grupo del sistema.
groupmod	Modifica un grupo ya existente.
grpck	Checa la integridad de los archivos de grupo etc/group y etc/gshadow.
grpconv	Crea archivo /etc/gshadow a partir de /etc/group.
grpuncov	Opuesto al anterior.
Id	Despliega información del usuario.
newgrp	Cambia el usuario primario.
passwd	Modifica la contraseña del usuario.
pwck	Comprueba la consistencia de los archivos /etc/passwd y /etc/shadow.
pwconv	Convierte a archivos de contraseñas ocultas los archivos de contraseñas normales, crea /etc/shadow a partir de /etc/passwd.
pwunconv	Opuesto al anterior.
useradd	Agrega un nuevo usuario al sistema.
userdel	Elimina un usuario del sistema.
usermod	Modifica un usuario ya existente.
vigr	Edita el archivo /etc/group.
vipw	Edita el archivo /etc/passwd.

Archivos generales de inicialización.

Ubicación	Descripción
/etc/profile	Contiene la configuración del perfil de arranque del *login*. Se ejecuta cada vez que el usuario ingresa al sistema.
/etc/bashrc	Contiene funciones de configuración como el umask, PS1 del *prompt*. Se ejecuta cada vez que se invoca al *Shell*.
/etc/motd	Mensaje del día para todos los usuarios, será mostrado al inicio de la sesión.
/etc/default/useradd	Configuración de los valores predeterminados al crear un usuario; directorio principal y grupo principal.
/etc/login.defs	Configuración de los valores predeterminados al crear un usuario; número de usuario y valores de la contraseña.
/etc/issue	Contiene el *banner* que se mostrará en el momento del *login* local.
groupdel	Elimina un grupo del sistema.
/etc/issue.net	Contiene el *banner* que se mostrará en el momento del *login* remoto, por ejemplo “telnet”.

11. Permisos

Linux cuenta con un sistema de permisos para la manipulación de archivos y directorios. Cada archivo/directorio tiene un propietario y está asignado a un grupo (al que normalmente pertenece el propietario). Cada archivo/directorio posee tres conjuntos de permisos que controlan el acceso de:

- El propietario.
- Miembros del grupo al que pertenece el archivo.
- Otros usuarios (*Other*).

Permisos básicos.

Tipo	Lectura/listado **r**	Escritura **w**	Ejecución acceso **x**
Archivo	Se puede leer el contenido del archivo.	Se puede cambiar el contenido del archivo.	Se puede ejecutar el archivo como un programa.
Directorio	Se puede listar el contenido del directorio.	Se pueden crear archivos en el directorio.	Se puede establecer ejecución de directorios.

Permisos especiales

Tipo	*set-uid* (su)	set-gid (sg)	*sticky* (t)
Archivo ejecutable	Ejecución con privilegios del propietario.	Ejecución con privilegios del grupo (no se utiliza)	Hacer residente (no se utiliza)
Directorio	No se utiliza	Crea nuevos archivos con el grupo del directorio padre.	Permiso de escritura sólo para el propietario del archivo.

Notación de permisos

Permiso	**Descripción**
r	Lectura
w	Escritura
x	Ejecución
s	set-uid
s	*Set-gid*
t	*sticky*

El formato de los permisos está signado de la siguiente forma:

- Los primeros tres permisos corresponden al usuario.
- Los siguientes tres permisos corresponden al grupo.
- Los últimos tres permisos corresponden al resto.

Antes del inicio de los permisos antecede el signo “**-**“.

Propietario	Grupo	Resto
- - -	- - -	- - -

Ejemplos:

Permiso de lectura: - r- - - - - - - - -

El propietario puede leer "**r",** el grupo y el resto no pueden leer

.

Permiso de escritura: - r – w r w – r - -

El propietario y el grupo pueden escribir y el resto no.

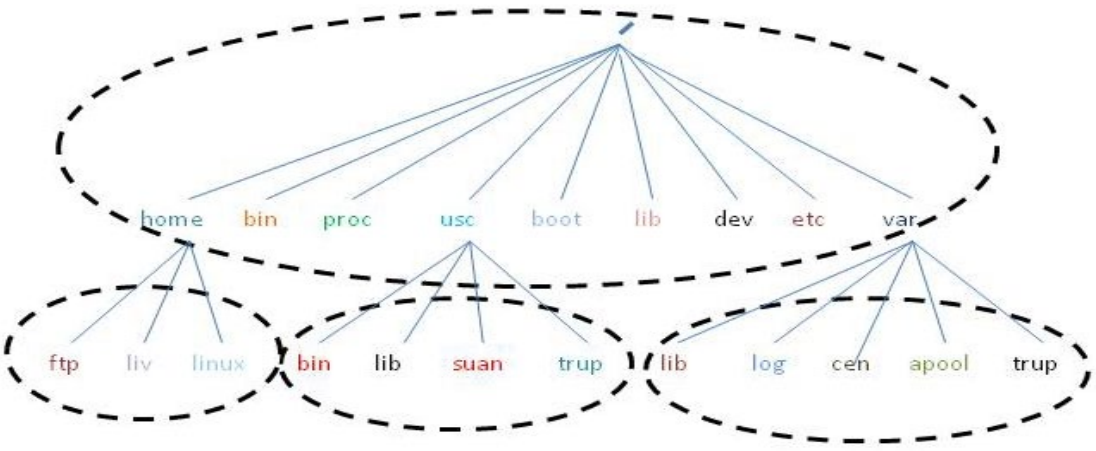
Permiso de ejecución: - **rwx r – x - - -**

El propietario y el grupo pueden ejecutar y el resto no.

Para ver los permisos que tienen un archivo o directorio se utiliza el comando **"ls-l"** el resultado muestra los permisos establecidos. Para cambiar los permisos, se utiliza el comando **"chmod"**.

3.3. Sistema de archivos

GNU/Linux está basado en el estándar de la jerarquía del sistema de archivos (Filesystem Hierarchy Estándar, FHS) el sistema de archivos reside bajo un árbol jerárquico de directorios, muy similar a los sistemas de archivos Unix. El directorio raíz es el primero y el único en el nivel superior del árbol jerárquico, lo que permite que sea menos propenso a errores y sea más apto para un fácil mantenimiento. El árbol de directorios completo puede ser dividido en partes más pequeñas que pueden estar en el disco duro o partición del mismo; las partes principales son los sistemas de archivos **raíz (/), /usr, /var y /home**. Cada directorio tiene un propósito especial, el árbol de directorios se ha diseñado para funcionar en una red de computadoras y pueden compartir algunas partes del sistema de archivos sobre dispositivos de sólo lectura (CD-ROM) o sobre la red a través de NFS (*Network File System*). Un ejemplo de la estructura de archivos a partir del directorio raíz *root* (/) se muestra en la siguiente ilustración.



Elaboración con base en: http://www.ibiblio.org/pub/linux/docs/LDP/system-admin-guide/translations/es/html/ch04.html,

Se tienen dos niveles de abstracción dentro del sistema de archivos:

Estructura física	Estructura lógica
Permite comprender cómo trabaja internamente el equipo.	Define la organización de los archivos y el árbol de directorios.

→ Tipos de archivos

GNU/Linux basa su sistema de archivos en los “nodos-i”. Un “nodo-i” es una estructura de control que contiene la información clave de un archivo necesario para el sistema operativo. Están formados por sectores del disco duro de 512 *bytes* que son utilizados por el sistema para indexar las direcciones de los bloques de datos de 4kb ocupados por un archivo, además de almacenar datos útiles para el sistema operativo, como el tamaño, el nombre o los permisos.

Este sistema aporta a GNU/Linux una ventaja importante con respecto a Windows, al no necesitar desfragmentarse, ya que en cualquier momento puede conocerse dónde se localiza la parte del archivo que se requiere a través del nodo-i. Existen diferentes tipos de archivos:

- **Archivos ordinarios.** Contienen la información con la que trabaja cada usuario y son archivos que contienen: texto, programas escritos por el usuario en algún lenguaje C, contienen caracteres ASCII y pueden ser creados, modificados, borrados, etcétera.

- Enlaces de archivos. Los enlaces son archivos que permiten que varios nombres se asocien a un archivo para que puedan tener varias instancias de un mismo archivo en diversos lugares de la estructura de archivos sin necesidad de copiaros, lo que permite tener una coherencia de los mismos y ahorran espacio en el disco duro, existen dos tipos de enlaces:

- **Enlaces físicos.** Los enlaces físicos también llamados enlaces duros representan un nombre alterno que se puede asignar a un archivo, esto hace que los nombres apunten físicamente al mismo sitio (nodo-i) del disco duro. Los dos nombres ocuparán diferentes lugares dentro del directorio, pero apuntan al mismo archivo. Si se borra uno de los dos, se borra la entrada y no el archivo. Si se borran ambos, desaparece la referencia dentro del directorio. Por ejemplo. Si dos usuarios necesitan compartir la información de un mismo archivo, las modificaciones que realice un usuario, no podrán ser utilizadas por el otro, ya que sólo se modificará la copia de uno de los usuarios. Si en lugar de tener una copia, cada uno de ellos utiliza un archivo tipo enlace al archivo original, cada vez que uno modifique su archivo, lo que estará modificando en realidad es su archivo común. Los enlaces físicos se crean con el comando **"ln"**; **ln nombre-real nombre-del-enlace-físico**.

- **Enlaces simbólicos (vínculos simbólicos).** Se utilizan para asignar más de un nombre a un archivo. Un vínculo simbólico es un archivo que sólo contiene el nombre de otro archivo. Cuando el sistema operativo opera sobre un vínculo simbólico, éste se dirige al archivo al que apunta el vínculo simbólico. A diferencia de los enlaces físicos, que existen dentro de la estructura de archivos, los enlaces simbólicos solamente hacen referencia al nombre de otro archivo. Se utiliza el parámetro **"-s"**. No sirven para directorios.

- **Directorios.** Son carpetas en donde se almacenan otros directorios denominados subdirectorios y archivos.

- **Archivos especiales.** Son archivos de bloque especial para dispositivos o *devices* físicos como unidades de almacenamiento, impresoras, terminales, etcétera. Linux los trata como archivos ordinarios, de manera que un usuario puede

abrir un archivo que está vinculado a una unidad y podrá leerlo, modificarlo, etcétera.

La siguiente tabla muestra los directorios posteriores a "*root*" [17].

Directorio	Descripción
bin	Guarda todos los comandos binarios esenciales.
boot	Son los archivos estáticos del cargador de arranque.
dev	Son los archivos que representan los periféricos.
etc	Guarda la configuración específica del sistema.
home	Directorio de usuarios.
lib	Son las librerías compartidas esenciales y módulos del kernel.
lost+found	Este archivo enlaza los enlaces perdidos de archivos recuperados.
media	Es el punto de montaje de medios extraíbles como USB, CD-ROM, etcétera.
mnt	Punto de montaje para sistemas de archivos temporales.
opt	Almacena todos los paquetes de *software*.
proc	Son todos los archivos virtuales de información para el kernel.
root	Directorio del superusuario.
sbin	Guarda los sistemas binarios esenciales.
srv	Guarda todos los datos de los servicios ofrecidos por el sistema.
sys	Tiene la misma función de *proc*, sólo que con otra organización.
tmp	Son todos los archivos temporales.
usr	Jerarquía secundaria; es decir, usuarios.
var	Guarda datos variables.
Windows	Si se tiene un Windows instalado deberá montarse en esta carpeta.

[17] Para mayor información, puedes consultar el siguiente sitio: http://server-die.alc.upv.es/alumno/linux/fsstnd12/fsstnd12-5.html

→ Tipos de sistemas de archivos

Los sistemas de archivos indican el modo en que se gestionarán los directorios dentro de las particiones; según su complejidad, tienen las siguientes características:

- Permiten acceder a la información de manera óptima.
- Garantizan la coherencia de la información.
- Permiten recuperar la información después de una caída severa.
- Permiten utilizar las listas de control de acceso (ACL).
- Agilizan las lecturas de datos (fragmentación).

Sistemas de archivos soportados por GNU/Linux:

1. Sistemas de archivos de **disco**: ext2, ext3, ReiserFS, JFS, XFS e ISO9660.
2. Sistemas en **red**: NFS (Network File System) y CIFS (Common Internet Files System).
3. Sistemas **virtuales**: VFS (Virtual File System) y SysFS.
4. Sistemas **especiales**: Swap y GmailFS.

Características:

- EXT2

Ha sido el sistema de archivos por *default* de Linux; además, es muy estable y rápido, aunque para archivos de gran tamaño es algo lento.

- EXT3

Es la versión mejorada de ext2, prevé la pérdida de datos por fallas de disco y de energía, también cuenta con la capacidad de mantener una bitácora del sistema, incluye tres modalidades para mantener las bitácoras: Journal,

Ordered y Write back. Actualmente es el más difundido dentro de la comunidad GNU/Linux.

- ReiserFS

ReiserFS utiliza algoritmos con árboles binarios (B-tree). Estos árboles binarios otorgan una solución más sencilla y eficiente al tratar con archivos pequeños.

- JFS

Este sistema cuenta con un árbol binario para guardar las direcciones de los archivos en el disco. Es ideal para necesidades muy altas de almacenamiento, se debe considerar JFS si se planea tener servidores con más de ocho terabytes de información. JFS es un sistema de 64 *bits*.

- XFS

De igual forma que el JFS el XFS es un sistema de 64 *bits* y al igual que ReirserFS utiliza algoritmos avanzados con árboles binarios. VFAT/NTFS.

- ISO9660

Sistema de archivos estándar para volúmenes de sólo lectura como el CD-ROM.

- NFS

Sistema por *default* para sistemas de archivos en red de GNU/Linux.

- CIFS

También conocido como SMB (Samba), permite compartir archivos e impresoras de sistemas Windows y Linux en la misma red de computadoras.

- VFS

Sistema de archivos virtual que proporciona una interfaz entre el núcleo del sistema operativo y el sistema de archivos real, para que las aplicaciones puedan acceder a la información sin importar qué sistema de archivos está almacenado en el disco. Los escritorios KDE y GNOME implementan sus propios sistemas de archivos virtuales a través de KIO y GNOME VFS, respectivamente.

- SysFS

Sistema de archivos virtual que provee el kernel 2.6, este proporciona información de los elementos de *hardware* y sus controladores para que el usuario pueda configurarlos en algunos de sus parámetros.

- SWAP

Sistema de archivos para el intercambio de procesos en el espacio de un disco (directorio o partición) cuando no se utilizan o no caben en la memoria física RAM.

- GmailFS for Linux

Provee a los usuarios acceso al sistema de correo Gmail de manera accesible.

También conocido como SMB (Samba), permite compartir archivos e impresoras de sistemas Windows y Linux en la misma red de área local (LAN).

Sistema por *default* para sistemas de archivos en red de GNU/Linux.

- VFAT (Virtual File Allocation Table) y NTFS (New Technology File System)

  Son sistemas de archivos principalmente ocupados por Microsoft. Linux puede leer archivos NTFS; pero aún no puede escribir en ellos. Sin embargo es posible escribir en FAT32 o FAT16, son totalmente seguros y estables.

- SWAP

  Es un sistema de archivos para la partición de intercambio de Linux. Todos los sistemas Linux necesitan una partición de este tipo para cargar los programas y no saturar la memoria RAM cuando se excede su capacidad.

Después de conocer las características y elegir el sistema de archivos, sigue la etapa de instalación del mismo, este tema será tratado en la unidad 7 de este material.

RESUMEN DE LA UNIDAD

GNU/Linux está regido por lo que se conoce como la Licencia Pública General de GNU, o GPL, General Public Licence. La GPL fue desarrollada para el proyecto GNU por la *Free Software Foundation*. Linux fue diseñado de tal modo que permite al administrador del sistema conocer la información que necesita sobre la máquina administrada y esto lo puede realizar a través de diversos comandos y programas descritos en esta unidad. Linux basa su sistema de archivos en los nodos-i, los cuales son sectores del disco duro de 512 *bytes* que son utilizados por el sistema para indexar las direcciones de los bloques de datos de 4kb ocupados por un archivo; además de almacenar datos útiles para el sistema operativo, como el tamaño, el nombre o los permisos.

Además Linux cuenta con un sistema de permisos que concederá o denegará la manipulación de archivos y directorios, lo cual hace que sea un sistema bastante seguro de operar. Su estabilidad, seguridad y rapidez han hecho que por excelencia éste sea el sistema operativo que muchos usuarios desean manejar para uso de servidores, computadoras, estaciones de trabajo, etcétera.

BIBLIOGRAFÍA

SUGERIDA

Autor	Capítulo	Páginas
Dávila, Manuel	1	1-42
	3	92-134

UNIDAD 4

FREE BSD

SUAyED UNAM
Sistema Universidad Abierta y Educación a Distancia

Plan 2012 2016 actualizado

OBJETIVO PARTICULAR

El alumno conocerá las características principales, operación y capacidad del sistema operativo FreeBSD.

TEMARIO DETALLADO

(8 horas)

4. Free/BSD

4.1. Características generales
4.2. Administración del sistema
4.3. Sistema de archivos

INTRODUCCIÓN

FreeBSD es un sistema operativo multiusuario de tipo UNIX (además de ser un *software* libre y gratuito[18]) para computadoras personales, basado en CPUs de arquitectura Intel que incluye procesadores 386, 486 y Pentium; también soporta procesadores compatibles con Intel como AMD y Cyrix. FreeBSD, está basado en el *release*[19] 4.4BSD-Lite del Computer Systems Research Group (CSRG) de la Universidad de California en Berkeley.

A lo largo de su historia, FreeBSD ha tenido diferentes *releases (versiones)*. El nombre de estas versiones puede ser de un número mayor o menor. El propósito de una versión mayor, es incluir nuevas funciones, y el propósito de una menor es corregir errores, mejorar el rendimiento y la estabilidad. Al añadir o quitar funciones a *FreeBSD* se puede perder compatibilidad con versiones anteriores, por lo que es muy importante mantener la compatibilidad entre *releases* menores.

En la presente unidad se abordan los aspectos más importantes de FreeBSD con el propósito de que conozcas qué tipo de aplicaciones puede ejecutar, en qué entornos puede emplearse y los servicios que ofrece.

[18] Información del copyright de FreeBSD, consultado el 10 de septiembre de 2012 en http://www.freebsd.org/es/copyright/index.html
[19] *Release*: nueva versión de una aplicación informática.

4.1. Características generales

FreeBSD[20] tiene las siguientes características:

Multitasking	Para asegurar la mejor compartición de recursos entre las aplicaciones y los usuarios
Multiusuario	Permite el acceso a múltiples usuarios.
Periféricos	Permite que todos los usuarios del sistema compartan estos recursos.
Conectividad	TCP/IP. Un equipo *FreeBSD* puede comunicarse con otros sistemas, también incluye soporte para los protocolos de comunicación: SLIP, PPP, NFS y NIS.
Protección de memoria	En caso de falla, asegura que las aplicaciones y usuarios no interfieran unos con otros.
Diseño	Diseñado desde su inicio como un sistema operativo de 32 *bits*.
X Windows (X11R6)	Provee una interfaz gráfica del usuario (GUI) para tarjetas VGA.
Compatibilidad binaria	Con programas nativos de SCO, BSDI, *NetBSD*, Linux y 386BSD.
Portabilidad	Aplicaciones disponibles en internet fáciles de portar.
Memoria virtual	Paginada bajo demanda para satisfacer eficientemente las aplicaciones.
Librerías	Compartidas para un uso eficiente del uso de disco y memoria equivalentes a UNIX y Windows.

[20] Acerca del proyecto FreeBSD, en http://www.freebsd.org/doc/es/books/handbook/nutshell.html

Herramientas de desarrollo	Lenguajes C, C++, Fortran, y otros.
Código fuente disponible	Código completo del sistema que ofrece un máximo control del entorno.
Plataformas soportadas	Soporta las siguientes arquitecturas; Intel (i386), *alpha*, amd64, ia64, i386, pc98, Sparck64®. También existen proyectos para soportar; ARM®, MIPS® y PowerPC®.
Framework integrados	Permite adaptar el entorno *FreeBSD* a necesidades de red específicas al soportar algunos módulos ya incluidos: ISDN, ATM, *EtherChanel*, *Frame Relay*, HDLC, PPPoE, L2TP.
Framework GEOM	Permite gestionar las particiones de lectura/escritura en discos a través del arreglo de discos RAID, también asigna protección criptográfica a los datos almacenados.
GBDE	**(**GEOM *Based Disk Encryption*) ofrece una protección criptográfica fuerte que puede utilizarse en sistemas de archivos y unidades *swap* entre otros tipos de unidades de almacenamiento.
MAC	(*Mandatory Access Control*) ofrece un control de acceso a archivos, es muy configurable.
PAM	(*Pluggable Authentication Modules*). Con PAM el administrador puede reforzar el modelo tradicional de autentificación usuario/contraseña. *FreeBSD* dispone de módulos para integrar PAM en una amplia gama de mecanismos de autenticación: Kerberos 5, OPIE, RADIUS, TACACS+.
Seguridad	*FreeBSD* trae integrados 3 firewalls con soporte NAT (*Network Translation Network*) para la creación de políticas de seguridad.

FreeBSD también cuenta con características más avanzadas que hacen posible un alto rendimiento, compatibilidad con otros sistemas y una mejor administración del sistema, a través de:

- *Bounce buffering*. Trata sobre la limitación en la arquitectura ISA de las PC que limitan el acceso directo a la memoria en los primeros 16 megabytes. Sistemas con más de 16 megabytes operan más eficientemente con periféricos DMA en el bus ISA.

- Buffer de *caché*. El buffer de *caché* (conjunto de memoria virtual y sistema de archivos) continuamente ajusta la cantidad de memoria usada por los programas y el *caché* del disco, por lo que los programas reciben una excelente gestión de memoria y un alto rendimiento en los accesos a disco, liberando al administrador del sistema del trabajo de ajustar los tamaños de la memoria *caché*.

- Módulos de compatibilidad. Permiten la ejecución de programas de otros sistemas operativos en *FreeBSD*, incluso programas para Linux, SCO, *NetBSD* y BSDI. Esto permite que los usuarios no tengan que recompilar programas ya compilados por algunos de los sistemas compatibles, teniendo acceso a programas como las extensiones para BSDI de Microsoft, FrontPage Server o WordPerfect para SCO y Linux.

- Módulos de *kernel* de carga dinámica. Permiten tener acceso a nuevos sistemas de archivos, protocolos de red y emuladores de binarios en tiempo de ejecución, sin necesidad de generar un nuevo *kernel*. Lo anterior, permite a los desarrolladores de otras partes ofrecer subsistemas completos como módulos de *kernel,* sin necesidad de distribuir el código fuente o complejos procedimientos de instalación; además de ahorrar tiempo con esto.

- Librerías compartidas. *FreeBSD* utiliza un esquema avanzado de librerías compartidas que reducen el tamaño de los programas, lo que ahorra espacio en disco y memoria.

-- Ventajas de usar FreeBSD

FreeBSD es un verdadero sistema abierto con todo el código fuente, incluyendo el *kernel* de todos los *daemons* del sistema, programas y utilerías, lo cual permite modificar cualquier parte de *FreeBSD* para adaptarlo a las necesidades personales, de organización o corporativas.

Algunas aplicaciones en la que es utilizado *FreeBSD*:

- Servicios de internet. Muchos proveedores de internet (ISPs) encuentran en *FreeBSD* la plataforma ideal para ofrecer servicios WWW, News, FTP, Email y otros. *Software* "*Ready-to-run*" como el servidor web Apache o el servidor FTP *ProFTPD*.

- Estación de trabajo X Windows. Funciona adecuadamente desde terminales muy básicas hasta avanzados monitores X. También ofrece controladores nativos para *hardware* gráfico de alta potencia.

- *Networking*. Permite el filtrado de paquetes, resolución de rutas y nombres; es decir, con *FreeBSD* se puede implementar un excelente *firewall* de internet, servidor de *e-mail*, servidor de impresión, servidor PC/NFS y más.

- Desarrollo de *software.* Cuenta con una serie de herramientas de desarrollo, incluyendo el compilador y *debugger* de GNU C, C++, desarrollos de Java, Fortran, Pascal, Modula3, Perl, *Shell script*, etcétera.

- Navegación por la red. Funciona como una verdadera estación de trabajo de UNIX para navegar en la red.

- Educación e investigación. *FreeBSD* es una excelente herramienta de investigación porque incluye el código fuente completo, por lo que estudiantes e investigadores pueden beneficiarse de este sistema abierto y bien documentado.

4.2. Administración del sistema

Antes de describir los aspectos sobre la administración del sistema, es importante comentar que *FreeBSD*[21] puede ser instalado a través de varios medios como son: CD-ROM, *floppies*, cintas magnéticas, una partición MS-DOS y, si se cuenta con conexión de red, se realiza vía FTP anónimo o NFS.

Debido a las diversas opciones que existen para instalar este sistema, es recomendable que busques literatura específica al respecto, puede ser desde el sitio oficial o algún otro tutorial.[22]

Para que se tenga un mayor conocimiento acerca de la instalación de este sistema operativo, se sugiere se revise el video del siguiente vínculo.

https://www.youtube.com/watch?v=EjfBQz-an-8

[21] Ver: Manual de FreeBSD en El Proyecto FreeBSD (2012).
[22] Te recomiendo el sitio oficial de FreeBSD, en donde te indican los pasos que debes seguir y los requerimientos que debes tener en tu equipo para su instalación: http://www.freebsd.org/doc/es/books/handbook/install.html. Consultado el 07/01/2013.

Elabora un breve instructivo en el que definas los pasos que se realizan para la instalación de este sistema operativo.

Si consideras que el video que revisaste es insuficiente, consulta la página *Google books*, y revisa el título “La super guía de instalación y configuración fácil de Free BSD”, del autor Norbert R. Ibañes. Revisa el contenido y lee la información que te sea de utilidad.

Una vez que has instalado el sistema operativo, también se recomienda revisar la documentación completa en línea de *FreeBSD* tecleando el comando “*man*”, ya que te servirá como una guía rápida.

Nota: El manual se instala de manera automática durante la instalación del sistema y se puede ejecutar desde la línea de comandos como usuario normal o como *root*.

El manual en línea “*man*” está dividido en las siguientes secciones:

1. Comandos de usuario
2. Llamadas de sistema y números de error
3. Funciones en las librerías de C
4. *Drivers* de dispositivos
5. Formatos de archivos
6. Juegos y otras diversiones
7. Información variada
8. Mantenimiento del sistema y comandos de sistema

En algunos casos el mismo comando aparece en más de una sección del manual en línea. Por ejemplo, existe un comando de usuario `chmod` y una llamada de sistema

`chmod ()`. En este caso, se puede especificar al comando `man` cuál de ellos se quiere ver y se especifica de la siguiente forma:

```
% man 1 chmod
```

Éste mostrará la información del comando de usuario `chmod`**.** Las referencias a secciones particulares del manual en línea tradicionalmente se incluyen entre paréntesis en la documentación escrita, de manera que `chmod(1)` se refiere al comando de usuario `chmod y chmod(2)` se refiere a la llamada del sistema.

Esto es correcto si se conoce el nombre del comando y simplemente se quiere saber cómo usarlo; pero ¿qué pasa si no se recuerda el nombre del comando? Puedes usar **man** para buscar palabras en las descripciones de los comandos usando el parámetro **-k:**

```
% man -k mail
```

Con este comando obtendrás una lista de todos los comandos que contienen en su descripción la palabra **"mail".** Si se quiere saber qué hacen todos los comandos existentes en `/usr/bin?` se utiliza:

```
% cd /usr/bin; man -f *
```

o

```
% cd /usr/bin; whatis *
```

(Ambos realizan la misma función)

A pesar de que *FreeBSD* ya trae una variedad de herramientas en su instalación, también necesita de *software* adicional que ayude a potenciarlo. Para ello cuenta con dos tecnologías complementarias, una de ellas es la colección de *Ports*, y la otra es

packages (paquetes) binarios. Puede emplear cualquiera de las dos para instalar versiones actuales, ya sea de forma local o desde la red.

→ Instalación de packages

Los *packages* (paquetes) binarios son programas precompilados que se distribuyen en formatos `.tgz` y `.tbz` se pueden encontrar en CD-ROM, la estructura de los *packages* es similar a la de *ports*, en la que cada categoría de *software* tiene su propio directorio que funciona como un todo para conformar el sistema paquetes/*ports*. Existen varias opciones para su instalación. Por ejemplo, para instalar el *software* Apache 2.2 se ejecuta la siguiente instrucción desde la cuenta ***root***:

Modo local:

`# pkg_add /tmp/apache -2.2.6_2.tbz`

`# pkg_add -r apache22` (la opción –r descarga e instala con las dependencias); es decir, con los programas que necesite para su funcionamiento.

Modo remoto:

`# ftp -a` **`ftp2.FreeBSD.org`** (utilizando el protocolo ftp)

Se inicia un diálogo interactivo hasta que termina la transferencia del archivo.

Connected to **`ftp.FreeBSD.org`**

`# pkg _add 1sof-4.56.tgz` (instalación del paquete).

Nota: Se recomienda utilizar el manual de *FreeBSD* para la correcta instalación de paquetes.

→ Instalación de aplicaciones (colección de Ports).

Los *Ports* de *FreeBSD* son un manejador de paquetes del sistema que permiten compilar e instalar una gran cantidad de programas con el mínimo esfuerzo, ya que debido a las diferencias que existen entre los estándares abiertos, lograr que un programa funcione en una versión diferente de Unix puede ser tardado y complicado.

Con la colección de *Ports* el trabajo complicado ya está realizado y sólo se tiene que teclear el comando ***make install*** para tener un programa perfectamente instalado y funcionando.

Existen dos formas de obtener un Port de FreeBSD para un programa. El primero requiere el CD-ROM de FreeBSD, y el otro requiere una conexión a internet.

◆ *Compilar Ports desde CD-ROM*
Si el CD-ROM de FreeBSD está en el lector y montado en el directorio /cdrom, se puede compilar toda la colección de Ports sin problemas, ya que ésta encontraría los **"tarballs"** en: **`/cdrom/ports/distfiles/`** (si existen allí). Los **"*tarballs*"** son una colección de archivos montados en un solo archivo. La utilería "**tar"** se utiliza para combinar algunos archivos en uno solo para archivar o facilitar su distribución. Un *tarball* es muy parecido a los archivos WinZip (Windows). Los *tarball* tienen extensiones como: **`.tar.gz, .tar.bz2`** o bien **`.tgz,`** los cuales pueden contener archivos de código o binarios. Existen otras maneras de hacerlo desde el CD-ROM incluyendo las variables del archivo `/etc/make.conf` **Nota.** Para la instalación de aplicaciones, se recomienda apoyarse de un manual, visita la siguiente página: **Instalando aplicaciones: la colección de Ports** http://people.freebsd.org/~jesusr/handbook/ports.html

◆ *Compilar Ports desde internet*

Si no se cuenta con CD-ROM o si se requiere instalar la última versión de *Ports* o *Port* que interesa, se necesita bajar el esqueleto de ese *port*. Aquí es importante considerar que hay que tener instalado el kit de actualización apropiado para el *release* (versión) de FreeBSD. Para conocer el kit apropiado, visita la página web de Ports en http://www.freebsd.org/ports/.
 	
La clave para los esqueletos es que el servidor FTP de FreeBSD puede crear *tarballs* al momento. Por ejemplo:
 	
`# cd /usr/ports`
`# mkdir databases`
`# cd databases`
`# ftp ftp.freebsd.org`
 	
`> cd /pub/FreeBSD/ports/ports/databases`
`> get gnats.tar`
`> quit`
`# tar xf gnats.tar`
`# cd gnats`
`# make install`

Nota: considerar sólo como un ejemplo, en la práctica hay que revisar los comandos cuidadosamente.

Hemos hablado de esqueletos…; pero, ¿qué es un esqueleto? Veamos a qué nos referimos.

→ Esqueletos

Un esqueleto es una serie de archivos que le indican a FreeBSD cómo compilar e instalar un programa. Los esqueletos más importantes son:

Makefile	**Directorio *files***
Componente más importante del esqueleto que específica cómo debe compilarse e instalarse un *port*.	Archivo que contiene el *checksum* del *port* llamado “md5” y utiliza el algoritmo hash “md5” para calcular el *checksum* de los *ports.*

Directorio *patches*	**Directorio pkg**
Contiene los parches necesarios para que todo funcione correctamente bajo el sistema FreeBSD.	Contiene tres archivos importantes: • COMMENT: descripción corta del programa. • DESCR: descripción más detallada. • PLIST: lista todos los archivos creados de la instalación.

→ Arranque de FreeBSD

Al proceso de inicio y carga del sistema operativo se le conoce como “*bootstrap*” o mecanismo de arranque. El proceso de arranque de FreeBSD es muy flexible ya que puede configurarse para que permita seleccionar diferentes sistemas operativos, versiones o *kernels* instalados en el mismo equipo. En equipos con arquitecturas x86, el sistema básico de entrada/salida (BIOS) es el responsable de cargar el sistema operativo, el cual busca en el disco duro el Registro Maestro de Arranque (RMA). Si el disco del equipo cuenta con varios sistemas operativos instalados, entonces puede hacer uso de un RMA diferente que despliegue la lista de los sistemas operativos instalados y permita elegir cuál de ellos se cargue.

El mecanismo de arranque de FreeBSD está dividido en tres etapas:

- 1ª etapa. Es ejecutada por el RMA que pone al equipo en un estado específico para ejecutar la segunda etapa. Se ejecuta: **/boot/boot1**, en este archivo, se encuentra el etiquetador de discos de FreeBSD (*disklabel*), el cual almacena información de las particiones para localizar y ejecutar boot2.

- 2ª etapa. El equipo ejecuta funciones más avanzadas del RMA. Se ejecuta**: /boot/boot2**, es más sofisticado y comprende lo necesario del sistema de archivos de FreeBSD para localizar archivos en él y además provee una interfaz simple para seleccionar el *kernel* o cargador que deberá ejecutarse.

- 3ª etapa. Finaliza la carga del sistema operativo. Se ejecuta: **/boot/loader** (cargador de arranque) que utiliza una serie de instrucciones integradas de fácil uso, respaldado por un intérprete de comandos más poderoso, con instrucciones de mayor complejidad.

→ Cuentas y usuarios

Todos los accesos al sistema deben ser a partir de que un usuario escribe su *login* y contraseña y procede a realizar tareas en el mismo. Pero como hemos visto en las anteriores unidades, los usuarios no son los mismos, tienen diferentes características y permisos. La configuración de las cuentas de usuarios es una tarea importante en la administración de FreeBSD.

FreeBSD permite que varios usuarios utilicen el mismo equipo a través de una red de computadoras. Los usuarios deberán tener una cuenta de acceso que contendrá:

- **Nombre de usuario**. Serán únicos en el equipo, no deberá haber dos usuarios con el mismo nombre, tendrán no más de ocho caracteres de longitud, en minúsculas.

- **Contraseña**. Cada cuenta deberá tener una contraseña asociada.

- **Identificador de usuario (UID**). El UID de usuario es un número entre 0 y 65536; se pueden tener varias cuentas con nombres de usuarios distintos.

- **Identificador de grupo (GID).** El GID de grupo es un número entre 0 y 65536 que identifica el grupo principal al cual pertenece el usuario. Los grupos son un mecanismo para controlar el acceso a los recursos del sistema con base en el GID y no en el UID. Un usuario puede pertenecer a más de un grupo.

- **Clase de *login***. Son una extensión al mecanismo de grupos que permiten mayor flexibilidad para adaptar el sistema a distintos usuarios.

- **Tiempo de cambio de contraseña**. Permite requerir a determinados usuarios o a todos cambiar sus contraseñas en determinado tiempo. Por *default* FreeBSD no obliga a cambiar las contraseñas de manera periódica. Esto es de ayuda para cuestiones de seguridad y de identidad personal.

- **Tiempo de expiración de cuentas**. Permite definir un tiempo limitado de vida de las cuentas, una vez vencido el tiempo de vida, la cuenta no puede ser utilizada para entrar al sistema. Los directorios y archivos serán conservados. Por *default* las cuentas en FreeBSD no expiran.

- **Nombre de usuario completo**. El nombre de usuario no identifica el verdadero nombre del mismo, el nombre real deberá ser asociado a la cuenta.

- **Directorio home**. El directorio "*home*" indica la ruta completa en el que el usuario se encontrará cuando acceda al sistema. Una práctica común es incluir todos los directorios *home* en: **`/home/nombre_de_usuario`** o bien en **`/usr/home/nombre_de_usuario.`**

- ***Shell* de usuario**. El *shell* provee el entorno por *default* mediante el cual los usuarios interactúan con el sistema. Existen distintos tipos de *shell* que los usuarios experimentados pueden configurar.

Existen de manera general tres tipos de cuentas: superusuario, usuarios del sistema y usuarios comunes:

Cuenta de superusuario (*root*)	Viene pre-configurada para facilitar la administración del sistema y no debe ser utilizada para actividades comunes, tales como el uso del correo electrónico, exploración del sistema o tareas de programación. La cuenta *root* puede operar sin límites y su mal uso o errores pueden ocasionar daños severos al sistema.
Cuentas del sistema	Este tipo de usuarios pueden ejecutar diversos servicios, tales como: TCP/IP, DNS, Web, Correo electrónico, etcétera. Algunos usuarios del sistema son: *operator, bind, daemon, news, httpd*, etcétera.
Cuentas comunes	Este tipo de cuentas, aíslan al usuario del entorno que impide que pueda dañar al sistema o a otros usuarios. Cada usuario puede configurar su entorno por medio de *shells*, idiomas, editores, etcétera.

Modificación de cuentas

Existen diversos comandos para poder modificar cuentas de usuario, los más comunes son:

- ***adduser***. Agregar nuevos usuarios.
- ***rmuser***. Eliminar usuarios.

- ***chpass***. Modificar la base de datos de usuarios.
- ***passwd***. Cambiar contraseña de usuario.
- ***pw***. Modificar aspectos de las cuentas de usuario.

Limitar usuarios

Por cuestiones de seguridad se deben limitar las capacidades y recursos que utilizan los usuarios, siendo éstas:

- ***Cuotas de disco***. Espacio en disco asignado al usuario.
- Uso de CPU y otros recursos que el usuario requiera.

Algunos comandos para limitar los recursos son:

- ***coredumpsize***. Espacio en disco.
- ***cputime***. Cantidad de tiempo de CPU que los procesos pueden consumir.
- ***filesize***. Tamaño máximo que puede tener un directorio.
- ***maxproc***. Número máximo de procesos que se pueden ejecutar a la vez.
- ***memorylocked***. Cantidad máxima de memoria que un proceso puede haber solicitado tener bloqueado en memoria principal.
- ***memoryuse***. Mayor cantidad de memoria que un proceso puede consumir en todo momento.
- ***openfiles***. Cantidad máxima de archivos que un proceso puede tener abierto.
- ***sbsize***. Cantidad límite de memoria de red que se puede consumir.
- ***stacksize***. Tamaño máximo que puede alcanzar la pila de un proceso.

→ Grupos

Un grupo es una lista de usuarios, se identifican por su nombre de grupo y GID (ID de grupo). En FreeBDS y en la mayoría de los sistemas Unix, los factores que toma en cuenta el núcleo para decidir si un proceso puede realizar algo es su ID de usuario y la lista de grupos a los que pertenece. A diferencia del ID de usuario, un

proceso tiene una lista de grupos asociados. Se pueden encontrar menciones al “ID de grupo” de un usuario o de un proceso, la mayoría de las veces se referirán al primero de los grupos de la lista. La correspondencia entre nombres e IDs de grupo se encuentran en: **`/etc/group`**.

→ Seguridad

La seguridad es una actividad que recae directamente en el administrador del sistema, aunque los sistemas FreeBSD y Unix cuentan con una seguridad inherente, el administrador deberá construir y mantener mecanismos de seguridad adicionales para evitar que los usuarios dañen el sistema. Los sistemas pueden ser tan seguros como uno los haga y siempre la seguridad va a competir con la comodidad a la que tendemos las personas. La seguridad, bien entendida, deberá implementarse en capas y monitorearse continuamente en busca de vulnerabilidades o intrusos, tampoco deberá exagerarse en su implementación, ya que afectaría el desempeño del equipo, acceso a las aplicaciones y usuarios.

Los problemas de seguridad se pueden dividir en diferentes categorías:

- Ataques de denegación de servicios (DoS).
- Comprometer cuentas de usuarios.
- Comprometer *root* a través de servidores accesibles.
- Comprometer *root* a través de cuentas de usuario.
- Creación de puertas traseras ("Backdoors").

Los ataques de denegación del servicio (DoS) son acciones que afectan a los sistemas en su funcionamiento normal, se realizan a través de “ataques de fuerza bruta” que sobrecarga la capacidad de los equipos o los inutiliza; algunos ataques aprovechan la debilidad de la *suite* de protocolos TCP/IP. Muchos ataques pueden solucionarse configurando las opciones apropiadas para limitar la carga al sistema operativo, los ataques a los protocolos de red TCP/IP son más complejos, ya que se pueden enmascarar o saturar de tráfico la conexión de red. Comprometer las cuentas de usuario es una actividad muy común, ya que pueden ejecutar servidores estándar como ***telnetd, rlogind, rshd y ftpd***, estos servidores por *default* no funcionan a través de conexiones cifradas. El administrador del equipo deberá estar muy atento a la revisión y análisis de los *logs* (bitácoras) del sistema para detectar cualquier intento de acceso, direcciones IP sospechosas, errores, etcétera. También se debe proteger la cuenta de *root*, ya que si un atacante la conoce o tiene acceso a ella, ocasionará daños muy graves al sistema. Un modelo de seguridad multicapa debe considerar los siguientes elementos:

- Proteger *root* y cuentas administrativas.
- Proteger los servidores que se ejecuten como *root*, como los binarios `suid/sgid`
- Proteger las cuentas de usuario.
- Proteger el archivo que contiene las contraseñas.
- Proteger el kernel (núcleo) y el sistema de archivos.
- Proteger los protocolos TCP/IP y servicios de red.

Nota: Al implementar los mecanismos de seguridad en FreeBSD, se recomienda contar con cierta experiencia en el manejo del sistema operativo y apoyarse en un manual que detalle los comandos a utilizar.

4.3. Sistema de archivos

La unidad más pequeña que utiliza FreeBSD es el archivo. Los archivos se almacenan en directorios; un directorio puede estar vacío o contener cientos de archivos. A partir de esto, se construye una jerarquía de directorios, facilitando la organización y orden de los datos. Para referirnos a archivos o directorios, se sigue la siguiente estructura: el nombre del archivo o directorio, seguido por una diagonal "/", y el nombre del documento. Por ejemplo **`/usr/documento.txt`** en donde el archivo **documento.txt** está dentro del directorio **`/usr.`**

FreeBSD no utiliza letras de unidades u otro nombre de unidad o ruta, por lo que no se puede escribir, por ejemplo: **`c:/usr/documento.txt.`** En FreeBSD el sistema de archivos *raíz* inicia con el símbolo "/"; a partir de esto, se van agregando otros directorios sin importar cuántos discos se tengan instalados en el sistema FreeBSD, ya que cada directorio aparecerá como si fuera parte de un solo disco.

El directorio *raíz* contiene lo que se denomina "puntos de montaje", que consiste en un directorio del que se pueden derivar directorios adicionales en un sistema padre o directorio raíz (/). Los puntos de montaje estándar son: **`/usr, /var, /tmp y /cdrom`**, estos directorios se relacionan con entradas en: /etc/fstab que contiene una tabla que se utiliza como referencia al sistema y contiene los diferentes sistemas de archivos y sus respectivos puntos de montaje. La mayoría de los sistemas de archivos en **`/etc/fstab`** se montan automáticamente en el arranque del sistema a menos que se tenga la opción "noauto".

Los directorios más comunes son:

Directorio	Descripción
/	Directorio raíz del sistema de archivos.
/bin/	Utilerías de usuario para ambientes monousuario y multiusuario.
/boot/	Programas y archivos de configuración necesarios durante el arranque del sistema operativo.
/boot/defaults/	Archivos de configuración por omisión del arranque.
/dev/	Nodos de dispositivo.
/etc/	Archivos de configuración y «scripts» del sistema.
/etc/defaults/	Archivos de configuración por omisión del arranque.
/etc/mail/	Archivos de configuración para agentes de transporte de correo.
/etc/namedb/	Archivos de configuración de *named*.
/etc/periodic/	«*Scripts*» que se ejecutan diariamente, semanalmente y mensualmente.
/etc/ppp/	Archivos de configuración de ppp.
/mnt/	Directorio vacío utilizado por administradores de sistemas como punto de montaje temporal.
/proc/	Sistema de archivos de procesos.
/rescue/	Programas enlazados estáticamente para restauraciones de emergencia.
/root/	Directorio local para la cuenta *root*.
/sbin/	Programas del sistema y utilerías fundamentales de administración para ambientes monousuario y multiusuario.
/tmp/	Archivos temporales. El contenido de /tmp *NO* se conserva después de un reinicio del sistema
/usr/	La mayoría de utilerías y aplicaciones de usuario.
/usr/bin/	Aplicaciones comunes, herramientas de programación y otras aplicaciones.
/usr/include/	Archivos «*include*» estándar de C.
/usr/lib/	Bibliotecas.
/usr/libdata/	Archivos de datos con diversas funciones.
/usr/libexec/	*Dæmons* del sistema y utilerías del sistema (ejecutados por otros programas).

/usr/local/	Ejecutables locales, bibliotecas, etcétera. también se usa como destino por omisión de la infraestructura de *ports* de FreeBSD
/usr/obj/	Árbol destino dependiente de arquitectura resultado de la compilación del árbol /usr/src.
/usr/ports	Colección de *Ports* de FreeBSD (opcional).
/usr/sbin/	*Dæmons* del sistema y utilerías del sistema (ejecutados por usuarios del sistema).
/usr/share/	Archivos independientes de la arquitectura.
/usr/src/	Archivos fuente BSD y/o local.
/usr/X11R6/	Ejecutables de la distribución X11R6, bibliotecas, etcétera (opcional).
/var/	Archivos multipropósito de log, temporales, en tránsito y de «*spool*».
/var/log/	Diversos archivos de log del sistema.
/var/mail/	Archivos de buzones de correo de usuarios.
/var/spool/	Directorios diversos del sistema de *spool* de impresora y correo.
/var/tmp/	Archivos temporales. Suelen conservarse cuando se reinicia el sistema a menos que */var* esté basado en memoria.
/var/yp	Mapas NIS.

Para ver un ejemplo de árbol de directorios puedes consultar el siguiente sitio:

http://www.freebsd.org/doc/es/books/handbook/disk-organization.html

La imagen muestra un árbol de directorios a partir de *root* en el cual el sistema de archivos “C” podría montarse en el sistema de archivos “B”.

→ Ventajas de contar con múltiples sistemas de archivos

- El sistema raíz puede montarse como sólo lectura, así se evita borrar por error o editar algún dato importante.

- Optimización automática del esquema de archivos en un sistema de archivos, dependiendo de cómo se utiliza el sistema. Por ejemplo, un sistema que contenga muchos archivos pequeños tendrá una optimización diferente de uno que contenga menos archivos, pero más grandes.

- Los sistemas FreeBSD son muy robustos, en caso de fallas de energía eléctrica, se pueden restaurar con mayor facilidad, si se reparten los datos en múltiples sistemas de archivos, aunque siempre existe el riesgo de que en una caída de energía se dañe la estructura del sistema de archivos, en cuyo caso se tendrá que restaurar a partir de un respaldo completo.

→ **Ventajas de contar con un solo sistema de archivos**

- Los sistemas de archivos son de tamaño fijo. Cuando se instala FreeBSD se crea el sistema con un tamaño específico, si posteriormente se requieren particiones de mayor tamaño se tendrá que realizar un respaldo, crear el nuevo tamaño del sistema y restaurar los datos.

- Los sistemas de archivos están alojados en particiones, es importante comentar que en FreeBSD el término *partición* no significa lo mismo que en otros sistemas, por ejemplo: MS-DOS. En FreeBSD cada partición se identifica con una letra de la "**a**" a la "**h**", puede contener solamente un sistema de archivos que se define a través de punto de montaje en la jerarquía del sistema de archivos o por la letra de la partición en la que se encuentran alojados.

- FreeBSD utiliza el espacio disponible del disco como un espacio de intercambio (*swap*) lo que le brinda a FreeBSD memoria virtual. Lo anterior permite al sistema comportarse como si tuviera más memoria de la que realmente tiene. Cuando se agota la memoria disponible, el sistema mueve algunos datos que no se están utilizando en ese momento al espacio de intercambio (*swap*), cuando requiere de nuevo esos datos, los trae al lugar en el que estaban.

A continuación se muestran algunas particiones con ciertas convenciones establecidas.

Partición	Descripción
a	Normalmente se refiere al sistema de archivos raíz (/)
b	Normalmente contiene el espacio de intercambio (swap)
c	Se refiere al mismo tamaño del *"slice"* (partición del sistema)
d	Anteriormente tenía un uso especial, actualmente no lo tiene

Cada partición que contiene un sistema de archivos, se almacena en lo que FreeBSD reconoce como un "*slice*" (partición), existen números de *slice* que se refieren a un nombre de dispositivo, al que precede la letra "s" y un número que puede ser 1 u otro mayor. Existe la partición física y lógica y sólo pueden existir cuatro particiones físicas en un disco; pero puede haber particiones lógicas dentro de una física, conocidas como *slice* extendidas y se numeran a partir del número 5. Las particiones físicas y lógicas se designan con las letras de la "a" a la "h".

Ejemplos:

da0s1 Se refiere a la primera *slice* en la primera unidad SCSI (*Small Computer System Interface*) que se utiliza para la transferencia de datos entre periféricos en el cable *bus* de una computadora: DVD, CD, impresoras, escáneres, etc., de alto rendimiento.

ad0s5 La primera *slice* extendida en el primer disco de tipo IDE (*Integrated Drive Electronics*).

da0a La partición **a** en la primera unidad **da**.

ad1s3e La quinta partición en la tercera *slice* de la segunda unidad de disco IDE.

También cada disco en el sistema tiene su designación, el nombre inicia por un código que indica el tipo de disco, seguido por un número que indica qué disco es el que inicia con 0.

Ejemplos de nombre de disco, *slice* y partición:

ad0s1a	La primera partición (a) en la primera *slice* (s1) en el primer disco IDE (ad0).
da1s2e	La quinta partición (e) en la segunda *slice* (s2) en el segundo disco SCSI (da1).

Códigos de dispositivos de disco:

Código	**Significado**
ad	Disco ATAPI (IDE)
da	Disco de acceso directo SCSI
acd	CDROM ATAPI (IDE)
cd	CDROM SCSI
fd	Disquete (floppy)

FreeBSD soporta el sistema de archivos: NFS (*Network File System*) que permite compartir directorios y archivos a través de la red, en el cual los usuarios pueden acceder a archivos que se encuentran en equipos remotos de forma transparente, como si se tratara de archivos locales, también se pueden compartir en red dispositivos de almacenamiento: CDROM, unidades ZIP, etcétera. Asimismo, puede montar otros sistemas de archivos como ext2, fat32, cd9660, etcétera.

→ Montaje y desmontaje del sistema de archivos[23]

El sistema de archivos se visualiza como un árbol invertido con los directorios: **`/.`** **`/dev`**, **`/usr`** y todos los demás directorios en el directorio raíz son ramas, las cuales pueden tener sus propias ramas; por ejemplo: /usr/local y así sucesivamente. Por diversas razones se pueden alojar algunos directorios del sistema, en sistemas de archivos separados, por ejemplo, el directorio **`/var`** contiene directorios **`log/`**, **`spool/`** y varios tipos de archivos temporales que pueden llegar a desbordarse por falta de espacio en el disco, también se puede requerir alojar ciertos árboles de directorios en discos físicos separados o en discos virtuales como en el caso de montar NFS para utilizar un CDROM.

Lista de comandos:

- Archivo **fstab** durante el proceso de arranque los sistemas de archivos listados en /etc/fstab se montan automáticamente, a menos que estén listados con la opción “noauto”.
- El comando **mount** monta un sistema de archivos, de la siguiente forma:

```
# mount dispositivo punto-de-montaje
```

- El comando **umount** desmonta un sistema de archivos de la siguiente forma:

```
# umount nombre-de-dispositivo
```

[23] Se recomienda revisar las opciones que existen para el uso de estos comandos en http://www.freebsd.org/doc/es/books/handbook/mount-unmount.html

RESUMEN DE LA UNIDAD

FreeBSD es un sistema operativo tipo UNIX para computadoras personales basado en CPU de arquitectura Intel que incluye procesadores 386, 486 y Pentium; también soporta procesadores compatibles con Intel como AMD y Cirix.

Entre sus características resalta su diseño de 32 bits, multitarea, multiproceso, multiusuario, compatibilidad binaria con programas nativos de SCO, BSDI, NetBSD, Linux y 386BSD, cuenta con herramientas de desarrollo: Lenguajes C, C++, Fortran, etcétera.

Free-BSD puede ser instalado a través de varios medios como son: CD-ROM, *floppies*, cintas magnéticas, partición MS-DOS y a través de FTP o NFS. La colección de *Ports* de FreeBSD permite compilar e instalar una gran cantidad de programas con un mínimo esfuerzo. Un esqueleto es una serie de archivos que le indican a FreeBSD cómo compilar e instalar un programa.

El proceso de arranque de este sistema consta de tres etapas, es muy flexible y puede configurarse para que permita seleccionar diferentes sistemas operativos en el mismo equipo. Es muy importante aplicar los mecanismos de seguridad para el buen funcionamiento del sistema, la seguridad deberá aplicarse en capas y monitorearse continuamente para encontrar vulnerabilidades o intrusos.

Los principales problemas de seguridad que pueden presentarse están relacionados con:

- Ataques de denegación de servicio (DoS).
- Comprometer cuentas de usuarios.
- Comprometer *root* a través de servidores accesibles.
- Comprometer *root* a través de cuentas de usuario.
- Creación de puertas traseras (“*Backdoors*”).

La unidad más pequeña que utiliza FreeBSD es el archivo. Los archivos se almacenan en directorios, el que puede estar vacío o puede contener cientos de archivos, a partir de lo cual se construye una jerarquía de directorios. El directorio raíz contiene “puntos de montaje” que consiste en un directorio del que se pueden derivar directorios adicionales en un sistema padre. FreeBSD permite instalar un solo sistema o múltiples sistemas de archivos. También puede montar otros sistemas de archivos como NFS (Network File System) ext2, fat32, cd9660, etcétera.

FreeBSD tiene disponible un conjunto de aplicaciones a través de dos sistemas: *packages* y *ports*, que cuentan con más de 1000 programas cada uno. Los *packages* son aplicaciones ya compiladas y "*ready-to-run*" y los *ports* son un conjunto mínimo de archivos que indican a FreeBSD como compilar e instalar un programa.

BIBLIOGRAFÍA

SUGERIDA

Autor	Capítulo	Páginas
Silberschatz, Abraham.	20	621-665
Manual de FreeBSD	2, 3, 4, 8, 13, 14	
Tanenbaum, Andrew	10	681-756

UNIDAD 5

ADMINISTRACIÓN DE ARCHIVOS

SUAyED UNAM
Sistema Universidad Abierta y Educación a Distancia

Plan 2012 2016 actualizado

OBJETIVO PARTICULAR

El alumno identificará las características principales del sistema de administración de archivos y la manera en que impactan en las aplicaciones de los sistemas de cómputo.

TEMARIO DETALLADO

(8 horas)

5. Administración de archivos

5.1. Conceptos básicos de archivos
5.2. Directorios y nombres de archivos
5.3. Permisos
5.4 Los nodos-i de UNIX
5.5 Jerarquía de directorios
5.6 Administración de dispositivos de entrada y salida (E/S)
5.7 Copias de respaldo y comprensión de archivos
5.8 Mantenimiento al sistema de archivos

INTRODUCCIÓN

En la mayoría de las aplicaciones, el archivo es el elemento central que proporciona el mecanismo para el almacenamiento y el acceso en línea a datos y programas que pertenecen al sistema operativo y a todos los usuarios del sistema de cómputo. El sistema de archivos consta de dos partes: una colección de archivos (cada uno para el almacenamiento de datos relacionados), y una estructura de directorios que organiza y proporciona información acerca de todos los archivos en el sistema. En esta unidad se abordarán los conceptos de archivo, su estructura y la forma en que los organiza el sistema operativo.

5.1. Conceptos básicos de archivos

Una de las funciones esenciales en todos los sistemas operativos, es la administración de archivos. La manera en que se estructuran, se nombran, se tiene acceso a ellos, se utilizan, se protegen y se implementan, son puntos muy importantes en el diseño de un sistema operativo. El “Sistema de Archivos” es la parte del sistema operativo que se ocupa de los archivos y todas las funciones relacionadas a éstos. Los archivos, de acuerdo con Tanenbaum (2003), son un mecanismo de abstracción que permite almacenar información en el disco y leerla después. Esto se realiza de manera transparente para el usuario; es decir, el usuario no se entera de los detalles de cómo y en dónde es almacenada la información. Una de las características más importantes de la abstracción, es la forma en que se nombran los objetos que se manejan, cuando un proceso crea un archivo le asigna un nombre, cuando se termina el proceso el archivo sigue existiendo y otros programas pueden utilizarlo a través de su nombre.

Los archivos se pueden estructurar de varias maneras, las más comunes son:

1. Secuencia de *bytes*
   a. El archivo es una serie no estructurada de *bytes*.
   b. Posee máxima flexibilidad.
   c. El sistema operativo no sabe qué contiene el archivo.

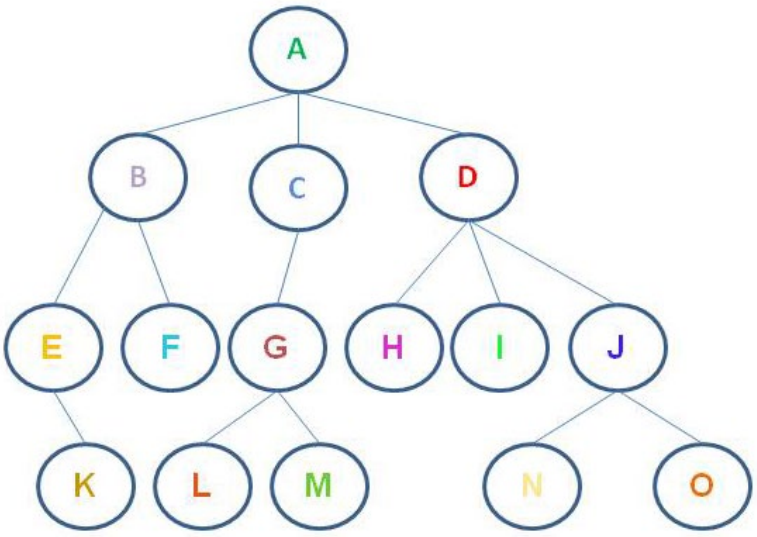
2. Secuencia de registros
   a. El archivo es una secuencia de registros de longitud fija, cada uno con su propia estructura interna.

3. Árbol

a. El archivo consta de un árbol de registros, no necesariamente de la misma longitud.

b. Cada registro tiene un campo llamado *key* (llave o clave) en una posición fija del registro.

c. El árbol se ordena mediante el campo de clave para permitir una rápida búsqueda de una clave particular. Observa el siguiente ejemplo:



Elaboración con base en
http://148.204.64.201/paginas%20anexas/POO/PRACTICA%2018.htm

La información de un archivo es definida por su creador. En un archivo se pueden almacenar diferentes tipos de información: programas fuente, programas objeto, programas ejecutables, datos numéricos, texto, registros de nómina, imágenes, grabaciones de sonido, etcétera, en diferentes formatos como TIF, WAV, MOV, PSD, WMV, MP3, PS, OTF, HTM, CSS, GIF y PPT.

Existen reglas concretas para nombrar archivos dependiendo del sistema operativo que se trate. Todos los sistemas actuales permiten utilizar cadenas de uno a ocho caracteres como nombres de archivo válidos. Por ejemplo: *Carlos*, *prueba* y *nómina*

son nombres de archivos válidos. También es posible que se permitan números o caracteres especiales del modo siguiente: carlos2, prueba!, nomina-2, que también son válidos. Algunos sistemas distinguen entre mayúsculas y minúsculas en los nombres como en el caso de Unix, en tanto que otros sistemas consideran los dos casos equivalentes, tal como MS-DOS. Varios sistemas manejan nombres de archivo de dos partes separadas por un punto, que indican algo acerca del archivo, como en el caso de MS-DOS, en donde el archivo prueba.c contiene una extensión que puede ser de uno a tres caracteres. También existen otros sistemas, como Unix, los cuales permiten más de dos extensiones, tal como nómina.c.Z con que se indica que el archivo nómina.c se comprimió utilizando un algoritmo de compresión Ziv-Lempel. Cuando se asigna un nombre a un archivo, éste se vuelve independiente del proceso del usuario, e incluso del sistema que lo creó.

Un archivo tiene generalmente los siguientes atributos:

1. Nombre	El nombre simbólico del archivo es la única información que se mantiene en forma legible para las personas.
2. Tipo	Esta información es necesaria para aquellos sistemas que soportan diferentes tipos de archivos: El nombre y una extensión, separados generalmente por un punto. Por ejemplo en MS-DOS un nombre puede tener hasta 8 caracteres, seguido por un punto y termina con una extensión; “.com”, “.exe” o “.bat”.
3. Ubicación	Esta información es un apuntador a un dispositivo y a la ubicación del archivo en dicho dispositivo. `C:\USER\DOCS\LETTER.TXT`
4. Tamaño	En este atributo se incluyen el tamaño actual del archivo (en *bytes*, palabras o bloques) y, posiblemente, el tamaño máximo permitido (como 600 Mb, 1Kb, 2G).

5.Protección	Información de control de acceso que determina quién puede leer, escribir, ejecutar, etcétera., el archivo.
6. Hora, fecha e identificación del usuario	Esta información puede mantenerse para: 1) la creación, 2) la última modificación y 3) el último uso. Estos datos pueden ser útiles para protección, seguridad y control de uso (Silbertschatz: 346-351).

Operaciones sobre los archivos (llamadas al sistema)

I. Crear un archivo.

Se debe encontrar espacio para el archivo en el sistema de archivos y posteriormente 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, por ejemplo: la llamada al sistema "***Create***" crea el archivo sin datos, el objetivo de la llamada anuncia que va a existir un nuevo archivo y establece algunos de sus atributos.

II. Escribir un archivo

Se hace una llamada al sistema **"*Write*"** especificando tanto el nombre del archivo como la información que se va a escribir en él. El sistema debe mantener un apuntador de escritura a la ubicación en el archivo donde va a tener lugar la siguiente escritura. El apuntador de escritura debe actualizarse siempre que ocurre una escritura. Si la posición actual es el fin del archivo, aumenta su tamaño, si la posición actual se encuentra en un punto intermedio del archivo, los datos existentes se sobrescribirán y se perderán.

III. Leer un archivo

Se hace una llamada al sistema **"*Read*"** 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. Tanto la operación de lectura como la de escritura emplean este mismo apuntador, ahorrando espacio y reduciendo la complejidad del sistema.

IV. Reposicionarse dentro de un archivo

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. La llamada se realiza con ***"Seek"*** (en archivos de acceso aleatorio).

V. Borrar un archivo

Se busca en el directorio el archivo designado. Una vez que se ha encontrado la entrada asociada, se libera todo el espacio del archivo (para que pueda ser reutilizado por otros archivos) y se borra la entrada del directorio. La llamada es con ***"Delete"***.

VI. Truncar un archivo

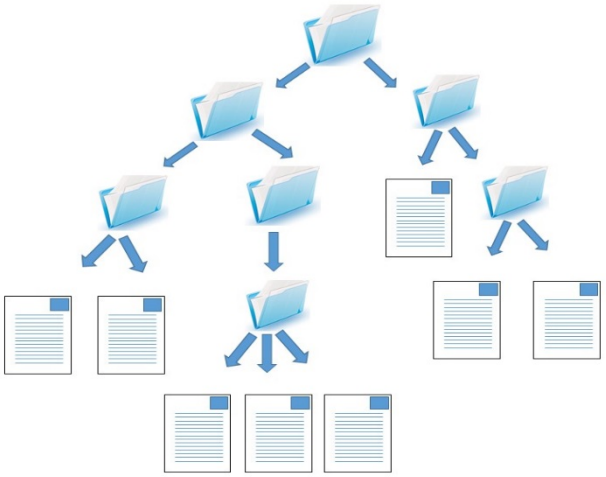
El usuario puede establecer o modificar los atributos de un archivo. Si desea que los atributos permanezcan iguales, pero quiere borrar el contenido del archivo. En lugar de obligar al usuario a borrar el archivo y después volver a crearlo, la llamada ***"Set attributes"*** permite que todos los atributos permanezcan sin modificación (excepto la longitud del archivo).

Tipos comunes de archivos

Como bien conoces, cada tipo de archivo tiene un tipo de extensión que lo caracteriza de acuerdo al programa en que es elaborado, por ejemplo, si se trata de un archivo ejecutable tendrá las extensiones .exe, .com, o .bin;
Si es de código fuente: C, cc, pas, java, asm, pl. O bien, si es de procesador de palabras: Txt, doc etcétera.

5.2. Directorios y nombres de archivos

Para llevar el control de los archivos, el sistema de archivos comúnmente está conformado por directorios o carpetas. La forma más sencilla es que un directorio contenga todos los archivos, y es conocido como directorio raíz, como se muestra como ejemplo en el siguiente esquema:



Elaboración con base en: http://bit.ly/1FT3yw1

Los sistemas de archivos de las computadoras pueden ser extensos. Algunos sistemas almacenan miles de archivos en cientos de gigabytes de disco, y para manejar todos estos datos se necesita cierta organización. De acuerdo con Silbertschatz (2002: 357-358), ésta generalmente se realiza en dos partes:

1. El sistema de archivos se descompone en particiones, también conocida como **minidiscos**. Cada disco en un sistema contiene por lo menos una partición. Algunos sistemas utilizan particiones para proporcionar varias áreas separadas dentro de un disco, tratando a cada una como un dispositivo de almacenamiento distinto, y otros sistemas permiten que las particiones sean más grandes que un disco de manera que puedan agrupar discos en una estructura lógica. De esta forma, el usuario sólo necesita preocuparse de la estructura lógica de directorios y archivos; puede ignorar completamente los problemas de la asignación física de espacio para los archivos. Por esta razón, las particiones pueden ser consideradas como discos virtuales.

2. Cada partición contiene información de los archivos dentro de ella. Esta información se mantiene en entradas en un **directorio del dispositivo** o **tabla de contenido del volumen**. El directorio del dispositivo (comúnmente conocido sólo como directorio) registra información –como nombre, ubicación, tamaño y tipo – de todos los archivos en dicha partición.

Un directorio puede ser visualizado como una tabla de símbolos que permiten traducir los nombres de los archivos y sus entradas en el directorio. El directorio puede ser organizado de diversas formas, con capacidad para poder insertar, borrar, etcétera, y buscar entradas con base en su nombre y obtener así una lista de todas las que se realizaron en él.

→ Nombres de archivos

Un nombre de archivo, según Flynn (2001), puede tener desde dos hasta varios componentes según el sistema de archivos que se utilice. Los dos componentes en la mayor parte de los nombres de archivo son: un **nombre** relativo (abreviado) del archivo y una **extensión.** El nombre relativo es el nombre seleccionado por el usuario cuando crea el archivo, por ejemplo, NÓMINAS, DIRECCIÓN, PROYECTO 2015, de forma general el nombre puede variar en longitud, uno o más caracteres especiales, e incluir letras y números. Sin embargo, todos los sistemas operativos tienen reglas que afectan la longitud del nombre relativo y de los tipos de caracteres especiales permitidos, por ejemplo MS-DOS acepta que se coloque en el nombre de uno a ocho caracteres alfanuméricos, sin espacios, en comparación con los sistemas operativos más actuales que permiten nombres con docenas de caracteres, incluyendo espacios. Al nombrar un archivo se debe considerar utilizar nombres relativos y descriptivos que permitan identificar con facilidad el contenido o propósito del archivo, para que sean fáciles de recordar y utilizar por otros usuarios del sistema. Por ejemplo, si el archivo se utiliza para ejecutar el programa

de control de inventario de teléfonos celulares, INVENTARIO, es un buen nombre, INVENTARIO DE CELULARES es mucho mejor.

La extensión tiene normalmente dos o más caracteres de longitud y está separada del nombre relativo por medio de un punto, esto es para identificar el tipo de archivo o su contenido. Por ejemplo, en MS-DOS un nombre con extensión sería INVENT.FOR, la extensión FOR le indica al sistema operativo que este archivo se escribió en el lenguaje de programación FORTRAN; otros ejemplos son: IMPUESTO.COB, que indica un archivo COBOL; AUTOEXEC.BAT es un archivo por lotes (*batch*), que se ejecuta de manera automática cuando el equipo arranca; INVENT.DAT indica que se trata de archivo que contiene datos utilizados con INVENT.FOR.

El sistema operativo reconoce algunas extensiones como FOR, BAS, BAT, COB y EXE, algunas extensiones se utilizan como señal para que el sistema utilice un compilador o programa en especial para ejecutar esos archivos; otras, como TXT, DOC, OUT, MIC y KEY, son creadas por aplicaciones o usuarios para su identificación. Siempre es recomendable utilizar el manual del sistema operativo antes de nombrar archivos, para que puedan seleccionar extensiones válidas y útiles. Existen otros componentes requeridos para el nombre completo de un archivo, pero depende del tipo de sistema operativo que se trate. Por ejemplo, el archivo llamado INVENT.FOR, con el sistema operativo MS-DOS, se identifica así: C:\PARTES\INVENT.FOR, que quiere decir que el archivo INVENT.FOR se encuentra en el directorio PARTES en la unidad C y requiere del compilador Fortran.

Usando el sistema operativo Unix, sería de este modo: /usr/imfst/flynn/invent.for.

Los mayoría de los sistemas operativos no requieren que los usuarios escriban nombres muy largos cada que se requiera acceder a un archivo, ya que el

administrador de archivos selecciona un directorio para el usuario cuando inicia la sesión interactiva en el equipo, por lo que todas las operaciones de archivo solicitadas por el usuario inician a partir de ese directorio "base". En este directorio el usuario selecciona un subdirectorio, que se conoce como directorio actual de trabajo y de ahí en adelante, se asume que todos los archivos se localizan en ese directorio actual. Para poder acceder a un archivo, el usuario escribe el nombre relativo y el administrador de archivos agrega el prefijo apropiado.

Al igual que en los archivos, hay ciertas operaciones que se realizan en los directorios, las más comunes son: buscar, crear, borrar, listar, renombrar, y recorrer.

→ Seguridad

Es muy importante considerar la confiabilidad y la protección de la información que se encuentra en un sistema de cómputo. La confiabilidad se puede proporcionar a través de copias duplicadas de archivos en diversos medios como discos, cintas, ópticos, etcétera, utilizando las diferentes técnicas de respaldo (*backup*) por día, semana y mes, para mantener una copia en caso de que el sistema de archivos sea destruido por error o de manera maliciosa. Los sistemas de archivos se pueden dañar por diversas situaciones como fallas de *hardware*, energía eléctrica, temperatura, vandalismos, ataques, etcétera. La protección puede aplicarse por seguridad física de los equipos, periféricos (discos), alarmas, etcétera.

5.3. Permisos

El acceso a los archivos debe limitarse de acciones no permitidas por medio de operaciones que su dueño (usuario, sistema; etcétera) asignará de acuerdo a necesidades específicas y que el sistema operativo reconocerá para permitir o negar su acceso.

Las operaciones sobre los archivos son:

- **Leer** el archivo.
- ***Escribir*** en un archivo.
- ***Ejecutar*****:** Subir un archivo a la memoria y ejecutarlo.
- ***Anexar*** nueva información en el archivo.
- ***Borrar*** un archivo.
- ***Listar.*** *Muestra los atributos del archivo.*

La asignación de permisos puede complementarse con otras acciones como renombrar, copiar o editar un archivo. Existen diferentes mecanismos de protección que tienen ventajas y desventajas; por lo que se debe elegir el más apropiado de acuerdo a las necesidades requeridas y el sistema de cómputo que se utiliza. Un pequeño sistema de cómputo que sólo lo utilizan algunos usuarios que no requieran cierta protección, será muy diferente a otro sistema en un centro de datos, con grandes equipos que atiendan áreas estratégicas en el que seguramente las necesidades de protección serán muy diferentes.

Mecanismos de protección

→ Control de acceso

Flynn menciona que “Los primeros sistemas operativos no permitían que los usuarios compartieran archivos” (Flynn, 2001: 202). Por ejemplo, se requerían diez copias del compilador FORTRAN para dar servicio a diez usuarios FORTRAN. Los sistemas actuales sólo requieren una copia para dar servicio a todos los usuarios que lo requieran, también permiten compartir cualquier archivo de datos o programas propiedad del usuario o del sistema. Las ventajas de compartir son variadas: ahorro de espacio, sincronización de actualizaciones de datos (como cuando dos aplicaciones actualizan el mismo archivo de datos), mejora la eficiencia del uso de los recursos del sistema (ya que si dos archivos están compartidos en la memoria principal se reducen las operaciones de entrada/salida).

Cada sistema operativo tiene un método propio para controlar el acceso a los archivos, y son:

1. **Matriz de control de acceso**. Es un modelo conceptual que describe el estado de protección de manera precisa. La matriz describe los permisos de los sujetos (usuarios o procesos) sobre los objetos (archivos, directorios, memoria, programas, etcétera); es fácil de implementar, pero sólo trabaja bien con pocos archivos y usuarios. En esta matriz cada columna identifica un usuario, y cada renglón, un archivo. La intersección de renglón y columna contiene los derechos de acceso para dicho usuario a dicho archivo. En la implementación real, los *bits* uno y cero representan las letras:

RWED

R = acceso lectura

W = acceso escritura

E = acceso ejecución

D = acceso eliminación

- = No está permitido el acceso

Ejemplo:

	Usuario	Usuario 2	Usuario 3	Usuario 4	Usuario 5
Archivo 1	RWED	R-E-	- - - -	RWE-	- - E –
Archivo 2	RE	- - - -	- - - -	- - - -	RWED

El usuario 1 cuenta con acceso ilimitado al archivo 1; pero sólo se le permite leer y ejecutar el archivo 2 y se le niega el acceso a otros tres archivos.

2. **Listas de control de acceso**. Estas listas son una modificación de la matriz de control de acceso. Cada archivo se introduce en la lista y contiene los nombres de los usuarios autorizados y el tipo de acceso permitido. Para abreviar la lista, sólo se incluye a los que pueden utilizar el archivo y se agrupa bajo un encabezado global WORLD a quienes se les niega cualquier acceso. Este sistema requiere menos espacio de almacenamiento que una matriz de control de acceso.

Ejemplo:

Archivo	Acceso
Archivo 1	Usuario1 (RWED), Usuario2 (R- E-) Usuario5, WORLD (- - - -)

El usuario 1 cuenta con acceso ilimitado al archivo 1, pero el usuario 5 tiene la palabra WORLD seguido de cuatro guiones, que indica que el acceso está negado a otros usuarios.

3. **Listas de capacidad**. Estas listas muestran la información de control de acceso desde una perspectiva diferente ya que enumera los usuarios y los archivos a los cuales tiene acceso cada uno. Requiere menos espacio de almacenamiento que una matriz de control de acceso y es más fácil de mantener que una lista de control de acceso cuando se añaden o eliminan usuarios del sistema.

   Ejemplo:

Usuario	**Acceso**
Usuario1	Archivo1 (RWCD), Archivo4 (R-E-)
Usuario2	Archivo1 (R-E-), Archivo2 (R-E-), Archivo3 (RWED)
Usuario3	Archivo2 (R-E-)

   En el ejemplo anterior se muestran los archivos para cada usuario.

4. **Cerraduras.** Son palabras de control para la protección de archivos, es un método de control de acceso muy diferente. La cerradura es muy similar a una contraseña, pero protege un solo archivo, la contraseña protege el acceso al sistema. Cuando se crea un archivo el propietario puede protegerlo con una cerradura, que se almacena en el directorio, pero no se revela cuando se solicita un listado de dicho directorio, una vez protegido, el usuario debe de proporcionar la palabra correcta para tener acceso al archivo.

De los esquemas tratados en este punto, el uso más común es la lista de control de acceso; sin embargo, las listas de capacidad, en su desarrollo más reciente, están ganando popularidad, ya que pueden controlar el acceso a dispositivos y archivos.

Ejemplo UNIX

En Unix, la protección de directorios se maneja como la protección de archivos, en la que a cada subdirectorio están asociados tres campos: *propietario, grupo y universo.* Cada uno compuesto de tres *bits* **"rwx",** de tal manera que un usuario puede listar el contenido de un subdirectorio sólo si el *bit* **"r"** está prendido en el campo apropiado, o bien, puede cambiar su directorio actual a otro directorio sólo si el *bit* **"x"** asociado con el directorio está prendido en el campo apropiado. También existen permisos básicos de Unix, como: SETUID y SETGID, que son permisos de acceso que pueden asignarse a archivos y directorios. Se utilizan principalmente para permitir a los usuarios del sistema ejecutar binarios con privilegios elevados de manera temporal para realizar una tarea específica. Si un archivo tiene activado el *bit* "SETUID", se identifica con una "s" en un listado de la siguiente manera:

```
-rwsr-xr-x 1 root shadow 27920 ago 15 22:45
/usr/bin/passwd
```

El comando "Chattr" agrega atributos (*append only, immutable*, etcétera). El comando Posix ACLs (getfacl, setfacl) y NFSv4 ACLs flexibilizan las ACLs estándar, lo que posibilita dar permisos específicos a más de un grupo, etcétera.

5.4. Los nodos-i de UNIX

El método para llevar el control de qué bloques, pertenecen a qué archivos consiste en asociar a cada archivo una estructura de datos llamada `nodo-i` (nodo índice). La ventaja principal de este esquema es que el `nodo-i` sólo tiene que estar en la memoria cuando el archivo correspondiente está abierto.

El sistema de archivos UNIX tiene la forma de un árbol que nace en la dirección raíz, con la adición de enlaces para formar una gráfica acíclica dirigida. Una entrada de directorio UNIX contiene una entrada para cada archivo de ese directorio, estas entradas utilizan el esquema de `nodos-i`. Una entrada de directorio contiene dos campos: el nombre de archivo (14 *bytes*) y el `nodo-i` correspondiente a ese archivo (2 *bytes*). Estos parámetros limitan el número de archivos por sistema de archivos a 64k. Los `nodos-i` de UNIX contienen atributos tales como: tamaño del archivo, hora de creación, último acceso, última modificación, dueño, grupo, información de protección y una cuenta del número de entradas de directorio que apuntan al `nodo-i`.

Un nodo-i contiene la información para que un proceso pueda acceder a un archivo. Durante el proceso de arranque del sistema, el núcleo lee la lista de nodos del disco y carga una copia en memoria, conocida como “tabla de nodos-i”. Esta tabla contiene la misma información que la lista de nodos-i, además de la siguiente información:

- Si el nodo-i está bloqueado.
- Si existe algún proceso esperando a que el nodo-i quede desbloqueado.

- Si la copia del nodo-i que está en memoria es diferente de la que hay en disco.
- Si la copia de los datos del archivo que hay en memoria, es diferente de los datos que hay en el disco.
- El número de dispositivo lógico del sistema de archivos que contiene el archivo.
- El número del nodo-i.
- Punteros a otros nodos-i cargados en memoria.
- Un contador que indica el número de copias del nodo-i que están activas.

5.5. Jerarquía de directorios

De acuerdo a lo expuesto por Silberschatz (2002: 359-367), el sistema operativo ejecuta el control de los archivos por medio de directorios o carpetas que en muchos sistemas también son archivos; su número y organización varía según el sistema de archivos de que se trate. A continuación se describen las principales jerarquías.

Sistemas de directorios de un solo nivel

Es la forma más simple de un sistema de directorios y consiste en que el total de archivos están contenidos en un solo directorio que en ocasiones se le conoce como directorio raíz. Esta jerarquía es fácil de soportar y entender, sin embargo tienen una desventaja importante, ya que cuando aumenta el número de archivos o cuando existen múltiples usuarios se podrían asignar por error los mismos nombres para los archivos. Por ejemplo, si el usuario **X** crea un archivo llamado *nomina*, y posteriormente el usuario **Z** crea también un archivo con el mismo nombre, el archivo de **Z** sobrescribirá al de **X**, esto debido a que todos los archivos se encuentran en el mismo directorio, por lo que los archivos deberán tener nombres únicos. La solución a esta problemática consiste en crear directorios diferentes para cada usuario del sistema. Es importante mencionar que por estas

limitaciones, esta jerarquía ya no se utiliza en los sistemas operativos multiusuario; sin embargo, se utilizan en sistemas pequeños.

Sistemas de directorios de dos niveles

Para evitar conflictos cuando cada usuario del sistema, elige nombres iguales para los archivos, se diseñó la jerarquía de dos niveles, ésta consiste, en que los usuarios tengan sus propios directorios de archivos de usuario (*user file directory, UFD*). Cada UFD tiene una estructura similar, pero lista sólo los archivos de un usuario. Cuando comienza un trabajo de usuario o se conecta un usuario (*login*), se hace una búsqueda en el directorio de archivos maestro (*master file directory*, MFD). El MFD está indexado por el nombre de usuario o el número de cuenta de acceso, y cada entrada apunta al UFD para dicho usuario y puede tener acceso a sus propios directorios. Sin embargo, una extensión de este esquema podría permitir a los usuarios, el acceso a los archivos de otros usuarios, si indican de quién es el archivo que desean abrir. Por ejemplo, la instrucción *open* (“x”) indica una llamada al sistema para abrir un archivo llamado x en el directorio del usuario, y *open* (“pedro/x”) es una llamada para abrir el archivo x que está en el directorio de otro usuario llamado “Pedro”.

Sistemas de directorios jerárquicos

Los sistemas de dos niveles eliminan los conflictos de nombres de archivos entre usuarios; pero no satisface la necesidad de los usuarios que cuentan con una gran cantidad de archivos. Es común que los usuarios ordenen sus archivos de forma lógica. Por ejemplo, un usuario podría tener una gran cantidad de archivos que integran un material para un curso, una segunda colección podría estar integrada por la serie de tareas de los alumnos de un grupo, un tercer grupo podría ser la serie de archivos de otro curso y así sucesivamente; por lo que el usuario requiere de una jerarquía general o árbol de directorios flexibles para agrupar sus archivos

en categorías naturales. Las categorías naturales consisten en extender la estructura del directorio de forma jerárquica, en la que los usuarios crean sus propios subdirectorios de acuerdo a sus necesidades. El sistema MS-DOS, por ejemplo, está estructurado como un árbol. El árbol tiene un directorio raíz. Cada archivo del sistema tiene un nombre de ruta único.
Un directorio (o subdirectorio) contiene un conjunto de archivos o subdirectorios. Un directorio es simplemente otro archivo, pero es tratado en una forma especial. Todos los directorios tienen el mismo formato interno. Un *bit* en cada entrada del directorio define la entrada como un archivo (0) o como un subdirectorio (1). Ciertas llamadas especiales al sistema crean y borran directorios.

Con un sistema de directorios con estructura jerárquica o de árbol, los usuarios pueden tener acceso a los archivos de otros usuarios, además de sus propios archivos.

Directorios de gráfica acíclica.
Una estructura jerárquica o de árbol prohíbe que se compartan archivos o directorios. Una gráfica acíclica (gráfica sin ciclos) permite que los directorios tengan subdirectorios y archivos que puedan compartirse. Tanto el archivo como el subdirectorio pueden estar en dos directorios diferentes. Una gráfica acíclica generaliza de forma natural el esquema de directorios con estructura jerárquica. Cuando varios usuarios trabajan como equipo, todos los archivos que se van a compartir pueden integrarse juntos en un directorio. Cada uno de los directorios de archivos de usuario de todos los miembros del equipo contiene este directorio de archivos compartidos como un subdirectorio inclusive cuando exista sólo un usuario que requiera colocar sus archivos en subdirectorios diferentes.

Una estructura de directorios de gráfica acíclica es más flexible que una estructura sencilla de árbol, pero también es más compleja.

5.6. Administración de dispositivos de entrada y salida (E/S)

El administrador de dispositivos, según Flynn (2001: 150-180), es la parte del sistema operativo que administra los dispositivos periféricos del sistema y requiere mantener un equilibrio entre la oferta y demanda de su utilización, y, de esta forma, balancear el suministro finito de los dispositivos del sistema con la demanda infinita de los usuarios por el uso de los mismos. La administración de dispositivos comprende cuatro funciones básicas:

1. Controlar el estado de cada dispositivo: discos, unidades de respaldo, impresoras, terminales, etcétera).
2. Utilizar políticas preestablecidas para determinar qué proceso utilizará un dispositivo y durante cuánto tiempo.
3. Asignar el uso de dispositivos.
4. Desasignar su uso en dos niveles; procesos y trabajo. En el nivel de proceso, cuando se ejecute un comando de entrada/salida y el dispositivo se libera de forma temporal, en el nivel de trabajo, cuando se termina el trabajo que realizó el dispositivo y se libera de forma permanente.

Los dispositivos periféricos se pueden clasificar en tres clases:

Dedicados
Se asignan sólo a un trabajo a la vez y le sirven todo el tiempo que está activo, las unidades de cinta, las impresoras y los graficadores, requieren este tipo de asignación, ya que si se compartieran sería ineficiente y no se obtendrían de manera correcta los trabajos asignados a éstos.

Compartidos

Se pueden asignar a varios procesos. Por ejemplo, varios procesos pueden compartir al mismo tiempo una parte del disco o cualquier otro dispositivo de almacenamiento de acceso directo, al entrelazar las solicitudes, el administrador de dispositivos debe de controlar el entrelazamiento, manejando los conflictos cuando varios procesos necesitan leer un mismo archivo en el disco, esto lo realiza a través del manejo de políticas predeterminadas.

Virtuales
Son una combinación de los dos primeros, son dispositivos transformados en dispositivos compartidos. Por ejemplo, las impresoras (dispositivos dedicados) pueden compartirse a través del programa "*spooling*" que tiene la capacidad de manejar las solicitudes de impresión a través del disco duro, en el que se van asignados las tareas de acuerdo a la disponibilidad de la impresora. El programa "*spooling*" se utiliza con frecuencia con el fin de acelerar dispositivos dedicados de entrada/salida.

Se tiene que considerar que cada dispositivo periférico es diferente en velocidad y capacidad de compartir, totalmente:

- Velocidad de los datos (teclado, disco duro, módem, ratón, etcétera).
- Aplicaciones (utilidad que se le da a un dispositivo; disco de archivos, disco de aplicaciones).
- Complejidad de control (interfaz de impresora, interfaz de disco, etcétera).
- Unidad de transferencia (flujo de *bytes*, bloques de E/S a disco).
- Representación de los datos (codificación de datos, convenios de paridad).
- Condiciones de error (naturaleza de errores, consecuencias, etcétera).

→ Comunicación entre dispositivos

El administrador de dispositivos se apoya de varias características auxiliares para continuar funcionando de manera óptima ante las condiciones demandantes de un sistema de cómputo ocupado, tres son los problemas que el sistema tiene que resolver:

1. Requiere conocer qué componentes están ocupados y cuáles están libres.
2. Tener la capacidad de aceptar las solicitudes que llegan durante el tráfico de entrada/salida.
3. Aceptar la disparidad de velocidades entre el CPU y los dispositivos de entrada/salida.

El **primer** problema se resuelve estructurando la interacción entre las unidades de entrada/salida, los **dos últimos** problemas se resuelven colocando en memorias intermedias los registros y la cola de solicitudes.

Cada unidad en el subsistema de entrada/salida puede terminar su operación de forma independiente de las demás. Por ejemplo, después que un dispositivo ha empezado a escribir un registro y antes de que termine la tarea, se puede cortar la conexión entre el dispositivo y su controlador, de manera que el controlador pueda iniciar otra tarea de entrada/salida con otro dispositivo.

Existen diferentes elementos que intervienen para que tenga éxito la comunicación entre dispositivos:

- **Channel Status Word (CSW).** El CSW en una localidad predefinida en la memoria principal que contiene información referente al estado del canal en una operación de entrada/salida. En esta localidad reside una bandera de *hardware* formada por tres *bits* que prueba el CPU para conocer cuándo un dispositivo ha terminado la tarea. Cada *bit* representa uno de los componentes del subsistema de entrada/salida: uno para el canal: otro para

la unidad de control, y uno más para el dispositivo. Cada *bit* cambia de cero a uno para indicar que la unidad ha pasado de libre a ocupada.

- **Encuesta.** Este elemento utiliza una instrucción especial de máquina para probar la bandera que asegura que la trayectoria está libre para poder seguir adelante en la operación de entrada/salida.

- **Interrupciones**. Es más eficiente que la encuesta y determina la mejor acción ante una situación presente, ya que comúnmente las unidades causan interrupciones de entrada/salida.

- **Acceso directo a la memoria (DMA).** Es una técnica de entrada/salida que permite que una unidad de control tenga acceso directo a la memoria principal para que una vez que se ha iniciado la lectura o escritura, el resto de los datos se pueda transferir desde la memoria (o hacia ésta) sin la intervención del CPU. Para activar este proceso, el CPU manda suficiente información a la unidad de control para iniciar la transferencia de datos, el CPU puede seguir adelante con otra tarea, mientras que la unidad de control termina de manera independiente la transferencia. Los dispositivos de alta velocidad como los discos, utilizan este modo de transferencia de datos.

- **Buffers**. Son memorias intermedias que se utilizan para sincronizar mejor el movimiento de los datos entre los dispositivos de entrada/salida lentos y un CPU muy rápido. Los *buffers* son áreas temporales de almacenamiento que residen en localidades convenientes en el sistema, memoria principal, canales y unidades de control. Se utilizan para almacenar datos que se han leído de un dispositivo de entrada antes que los requiera el procesador y para almacenar datos en un dispositivo de salida. Un uso común de los *buffers* se da cuando los registros en bloques son leídos o escritos en un dispositivo de entrada/salida. Existe la técnica de doble *buffer* que se utiliza

para minimizar el tiempo ocioso de los dispositivos y también para maximizar su producción. Los dos *buffers* se encuentran en la memoria principal, en los canales y en las unidades de control. El objetivo principal del doble *buffer* consiste en tener un registro listo para su transferencia hacia la memoria o desde ésta en cualquier momento, con el fin de evitar todo retraso posible debido a la necesidad de esperar que un *buffer* se llene de datos, por lo que mientras el CPU procesa un registro, el canal puede leer o escribir otro.

→ Administración de las solicitudes de entrada/salida

El administrador de dispositivos divide la tarea en tres partes, cada una manejada por un componente específico de *software* del subsistema de entrada/salida:

1 **Controlador de tráfico de entrada/salida.** Observa el estado de los dispositivos, unidades de control y canales. Su trabajo se complica conforme aumentan las unidades en el subsistema de entrada/salida y las trayectorias entre estas unidades. Este controlador realiza tres tareas importantes:

- Establecer si existe por lo menos una trayectoria disponible.
- Si existe más de una, determina cuál selecciona.
-Si las trayectorias están ocupadas, define cuándo quedará disponible una.

Para realizar lo anterior, el controlador de tráfico guarda una base de datos que contiene el estado y las conexiones de cada unidad del subsistema de entrada/salida y los agrupa en bloques de control de canal, bloques de control de unidades de control y bloque de control de dispositivos.

2. **Planificador de entrada/salida.** Realiza la misma tarea que el planificador de proceso; es decir, asigna dispositivos, unidades de control y canales. En condiciones de trabajo pesado, cuando la cantidad de solicitudes rebasa las

trayectorias disponibles, el planificador de entrada/salida debe decidir qué solicitud se atiende primero.

3. **Manejador de dispositivos de entrada/salida**. Procesa las interrupciones de entrada/salida, maneja los errores y proporciona algoritmos detallados de programación que dependen de cada dispositivo; es decir, cada tipo de dispositivo de entrada/salida tiene su propio algoritmo de manejo de dispositivo.

→ Tipos de dispositivos

Dispositivos legibles por los humanos
Son apropiados para la comunicación con el usuario. *Ejemplo*: terminales de video, teclados, pantallas, impresoras, etcétera.

Dispositivos legibles por la máquina
Son adecuados para comunicarse con equipos electrónicos. *Ejemplo*: discos, unidades de cinta, sensores, controladores e impulsores.

Dispositivos de comunicaciones
Apropiados para comunicarse con dispositivos lejanos. *Ejemplo*: adaptadores de líneas digitales, módem.

5.7. Copias de respaldo y compresión de archivos

Como parte de la administración de un sistema de cómputo, se debe asegurar que los datos no se pierdan en caso de una falla del sistema operativo o la presencia de algún agente externo que afecte la información (virus, etcétera). A esta acción se le denomina *copias de seguridad*. Para esto podemos emplear programas de sistema que respalden datos del disco en otro dispositivo de almacenamiento, como un disco flexible, una cinta magnética o un disco óptico. La recuperación de la pérdida de un archivo individual, o de todo un disco, puede implicar simplemente restablecer los datos a partir del respaldo.

Para minimizar el copiado requerido, podemos utilizar la información de cada entrada del archivo en el directorio. Por ejemplo, si el programa de respaldo sabe cuándo se realizó el último respaldo de un archivo, y la fecha de la última escritura del archivo, en el directorio indica que el archivo no ha cambiado desde ese momento, entonces el archivo no necesita copiarse nuevamente.

Los tipos de copias de seguridad pueden ser:

- **Normal**

  Conocido como respaldo completo, se copian todos los archivos y directorios seleccionados. Este tipo de respaldo no toma en cuenta los marcadores (*bits*) para determinar qué archivos respaldarán, elimina el atributo de archivo de todos los archivos que se van a respaldar.

- **Copia**

  En este tipo de respaldo realiza una copia de seguridad de todos los archivos y directorios seleccionados y no se buscan ni se borran los marcadores.

- **Diferencial**

  Aquí sólo se realiza una copia de seguridad de los archivos y directorios seleccionados que tienen un marcador. Este tipo de respaldo es moderadamente rápido en la copia y restauración de datos.

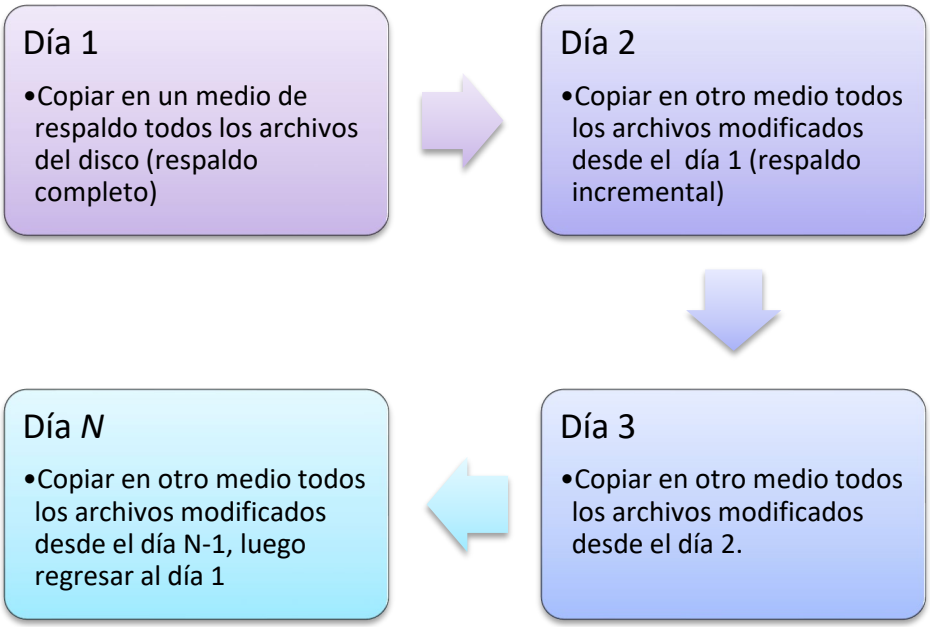
- **Incremental**

  En este sólo se realizan copias de seguridad de los archivos y directorios que tienen un marcador.

- **Diaria**

  Realiza copia de seguridad de todos los archivos y directorios seleccionados que han cambiado durante el día. Una copia de este tipo no busca ni borra los marcadores.

Ejemplo de un plan de respaldos:



El respaldo del nuevo ciclo puede escribirse sobre el conjunto anterior, o sobre un nuevo conjunto de medios de respaldo. Una ventaja de este ciclo de respaldo es que se puede restablecer cualquier archivo que se haya borrado accidentalmente durante el ciclo, recuperando el archivo borrado a partir del respaldo del día anterior. También es muy importante almacenar estos respaldos permanentes en un lugar alejado de los respaldos regulares, como protección contra peligros como un incendio que pudiera destruir el contenido de la computadora y también todos los respaldos.

Si en este proceso se reutilizan los medios, se debe tener mucho cuidado de no reutilizar estos medios demasiadas veces (desgaste físico), ya que puede no recuperarse el respaldo. (Meza, n.d: 85)

→ Compresión de datos

La compresión de datos es el proceso que permite reducir el tamaño de los archivos sin que se pierdan los datos originales, el nivel de compresión depende de los siguientes factores: tamaño de archivo, tipo, algoritmo o esquema de compresión. Los archivos tipo texto tienen un mejor nivel de compresión que los archivos de imágenes o de audio, ya que se basan en el idioma humano e incluyen altos índices de palabras repetidas, así como construcciones gramaticales y conjugaciones. Los archivos de imágenes o audio son poco propicios a la compresión, ya que contienen pocos patrones repetidos o porque su formato ya incluye cierta compresión como los archivos gráficos JPEG o los de audio MP3. De manera general, la compresión de datos busca encontrar en el texto original palabras o patrones repetidos para posteriormente guardar el dato junto al número de veces que se repiten. Por ejemplo, si en un archivo aparece una secuencia como "CCCCCC", ocupando 6 *bytes* se podría almacenar simplemente "6C" que ocupa sólo 2 *bytes*.

Algoritmos de compresión:

- Algoritmos que buscan series largas que se codifican en formas más reducidas.
- Algoritmos que examinan los caracteres más repetidos que se codifican de forma más corta (algoritmo de Huffman).
- Algoritmos que construyen un diccionario con los patrones encontrados, al cual se hace referencia posteriormente.

En la compresión hay que tomar en cuenta dos conceptos:

Redundancia	Entropía
Datos o patrones que son repetitivos o previsibles	Información esencial que se define como la diferencia entre la cantidad total de datos de un archivo y su redundancia.

Archivos compactados.

Además de los algoritmos de compresión, existen varios formatos de archivos; de aplicación general, orientados a imágenes, de audio y de video.
Para la compresión de archivos de cualquier tipo existe el formato BZIP2, este compresor es de uso libre, en donde un bloque de datos se convierte en otro que refleja la repetición de ciertas combinaciones. También existe GZIP de uso libre que considera el número de ocurrencias de un símbolo en una estructura jerárquica. El formato ZIP es una versión similar de GZIP desarrollado por la empresa PKWARE que ha implementado los programas Pkzip y Pkunzip. Existen otro tipo de programas que también realizan el proceso de compresión y son:

Programa	Extensión de archivo	Sistema operativo
WinZIP	.zip	Windows
WinRAR	.rar	Windows
PKzip	.zip	Windows
ARJ	.arj	Windows
Gzip	.gz	Linux
Stuffit	.sit	Mac OS

Para los archivos de audio el formato de compresión más común es MP3 que incluye una codificación de la información original que se registra como modulación de pulsos. Para las imágenes se utilizan los formatos GIF y JPEG. Para video los formatos más extendidos son: MPEG-1, MPEG-2 y MPEG-4.

5.8. Mantenimiento al sistema de archivos

Con el paso del tiempo, los archivos que se encuentran en el sistema crecen y disminuyen en tamaño, se crean nuevos y se borran otros, también se llevan a cabo constantes operaciones de lectura y escritura sobre varios de ellos. Aunado a lo anterior, en ocasiones existen fallas de energía eléctrica de manera intencional o por errores de operación de las computadoras, los cuales dañan la información, que en ese momento se trabajaba, afectando el funcionamiento de los discos duros y del sistema de archivos. Uno de los problemas más graves que afectan al sistema de archivos es que se altere su consistencia (Tanenbaum, 2003: 421-424), este problema se agudiza si algunos de los bloques que todavía no se han escrito en el disco son bloques de nodos-i, bloques de directorio o bloques que contienen la lista de espacio libre. Para resolver el problema de la inconsistencia, los sistemas operativos contienen utilerías que verifican la inconsistencia. Por ejemplo, UNIX tiene el programa "**`fsck`**" y Windows tiene "`scandisk`" o "`chkdsk`". Estos programas pueden ejecutarse cada vez que inicia el sistema operativo o bien después de una caída del sistema. En UNIX se pueden realizar dos tipos de verificaciones de consistencia: de bloques y de archivos. Para probar la consistencia de los bloques el programa "`fsck`" construye dos tablas, en la que cada tabla contiene un contador para cada bloque, que en un inicio se establece en

0. Los contadores de la primera tabla cuentan las veces que cada bloque está en el archivo, y la segunda, las que cada bloque está presente en la lista libre (mapa de *bits* libres). Posteriormente, el programa lee el total de nodos-i. A partir de un nodo-i, es posible construir una lista del total de números de bloque empleados en el archivo correspondiente. Cada que se lee cada número de bloque, se incrementa el contador de la primera tabla. Posteriormente, el programa examina la lista libre para ubicar todos los bloques que no se están utilizando. Con cada aparición de un bloque en la lista libre hace que se incremente el contador de la segunda tabla. Si el sistema de archivos es consistente, todos los bloques tendrán un 1 en la primera tabla o en la segunda, pero no en ambas. Sin embargo, podrían existir otros errores como: bloques ausentes en ambas tablas (bloques faltantes), bloques que aparecen dos veces en la lista libre o tener un mismo bloque en dos o más archivos, estas fallas desperdician espacio en disco y reducen su capacidad. El programa “fsck” también examina el sistema de directorios, en este caso utiliza una tabla de contadores, sólo que éstos son de archivos, no de bloques, en el que a partir del directorio raíz, desciende en forma recurrente por el árbol, revisando cada directorio del sistema de archivos. Es importante recordar, que debido a los enlaces duros, un archivo podría ubicarse en dos o más directorios. Los enlaces simbólicos no cuentan y no se incrementa el contador del archivo. Al terminar la verificación, se genera una lista indexada por número de nodo-i, que indica cuántos directorios contiene cada archivo. Posteriormente compara esas cifras con las cuentas de enlaces almacenados en el nodo-i. Estas cuentas toman el valor 1 cuando se crea el archivo y se incrementan cada que se establece un enlace (duro) con el archivo. Cuando ambas cuentas coinciden el sistema de archivos es consistente.

En el caso de Windows Server y el sistema de archivos NTFS (Sistema de archivos de Windows NT) adopta un enfoque diferente para el mantenimiento y robustez del sistema de archivos, ya que en NTFS, todas las actualizaciones de las estructuras de datos del sistema de archivos se realizan dentro de transacciones. Esto es, antes de que se altere una estructura de datos, la transacción escribe un registro de

bitácora que contiene la información para rehacer y eliminar, después que la estructura se modificó y se agrega un registro en la bitácora para indicar que dicha transacción tuvo éxito. Después de una caída, el sistema puede restablecer las estructuras de datos del sistema de archivos a un estado de consistencia procesando los registros de la bitácora. Esto lo realiza en primer lugar rehaciendo las operaciones para transacciones que se alteraron y posteriormente eliminando las operaciones para transacciones que no se alteraron antes de la caída. Este esquema no garantiza que todos los contenidos de los archivos de los usuarios sean correctos después de la caída, sólo asegura que la estructura de datos del sistema de archivos no esté dañada y que reflejen un estado consistente anterior a la caída. Es posible extender el esquema de transacciones para proteger los archivos de los usuarios, sólo que esto afectará el desempeño del sistema. Windows server tiene la herramienta "**chkdsk**" que se puede utilizar para diagnosticar y corregir los problemas de disco, mejorar el rendimiento, e inclusive, comprimir datos. Para su utilización deberán cerrarse todas las aplicaciones y archivos que se estén ejecutando.

El mantenimiento al sistema de archivos también deberá considerar otras herramientas propias de cada sistema operativo o desarrollado por terceros para asegurar un funcionamiento correcto y óptimo del sistema; las herramientas más comunes son:

- Administración de particiones lógicas.
- Formato lógico de unidades lógicas.
- Conversión a otros sistemas de archivos.
- Diagnóstico y reparación de errores físicos en los discos.
- Desfragmentación del sistema de archivos.
- Eliminación de archivos temporales que no se utilicen.
- Generación de respaldos.
- Verificación y reparación de archivos importantes del sistema operativo.

RESUMEN DE LA UNIDAD

El “sistema de archivos” es la parte del sistema operativo que se ocupa de los archivos y las funciones relacionadas a éstos. Los archivos se pueden estructurar de varias maneras, las más comunes son: secuencia de *bytes*, secuencia de registros y tipo árbol.

Un archivo tiene generalmente los siguientes atributos:

1. Nombre
2. Tipo
3. Ubicación
4. Tamaño
5. Protección
6. Hora, fecha e identificación del usuario

Para llevar el control de los archivos, el sistema de archivos comúnmente tiene directorios o carpetas. Los sistemas de archivos de las computadoras pueden ser extensos. Algunos sistemas almacenan miles de archivos en cientos de *gigabytes* de disco. Para manejar todos estos datos, se necesitan organizar. Esta organización generalmente se realiza en dos partes: *particiones* y *volúmenes*.

Cada sistema operativo tiene un método para controlar el acceso a los archivos, tales como; las matrices de acceso, listas de control, listas de capacidad y las cerraduras. El acceso se permite o se niega dependiendo de varios factores, uno de los cuales es el tipo de acceso solicitado. El sistema operativo Unix utiliza método nodo-i para llevar el control de qué bloques pertenecen a qué archivos, el cual consiste en asociar a cada archivo una estructura de datos llamada nodo-i

(nodo índice). La ventaja principal de este esquema es que el nodo-i sólo tiene que estar en la memoria cuando el archivo correspondiente está abierto. El sistema de archivos UNIX tiene la forma de un árbol que nace en la dirección raíz, con la adición de enlaces para formar una gráfica acíclica dirigida. Una entrada de directorio UNIX contiene una entrada para cada archivo de ese directorio, estas entradas utilizan el esquema de nodos-i. Una entrada de directorio contiene dos campos: el nombre de archivo (14 *bytes*) y el $\texttt{nodo}^{-i}$ correspondiente a ese archivo (2 *bytes*).

El número y organización de directorios varía según el sistema y son: directorio de un solo nivel, directorio de dos niveles, directorios con estructura de árbol y directorios de gráfica acíclica.

El administrador de dispositivos es la parte del sistema operativo que administra los dispositivos periféricos del sistema y requiere mantener un equilibrio entre la oferta y demanda de su utilización, los dispositivos periféricos se clasifican en: dedicados, compartidos y virtuales.

Por otro lado, hay que asegurar que los datos no se pierdan en caso de una falla. Para esto, podemos emplear programas del sistema para respaldar datos del disco a otro dispositivo de almacenamiento, como un disco flexible, una cinta magnética o un disco óptico. La recuperación de la pérdida de un archivo individual, o de todo un disco, puede implicar simplemente restablecer los datos a partir del respaldo. Las copias de seguridad pueden ser de tipo: normal, copia completa, diferencial y diaria.

Finalmente, para que una computadora funcione correctamente y con un óptimo desempeño, es necesario realizar el mantenimiento del sistema de archivo a través de los programas “utilerías” que contienen los diferentes tipos de sistemas operativos, o bien, con utilerías desarrolladas por terceros.

BIBLIOGRAFÍA

SUGERIDA

Autor	Capítulo	Páginas
Silbertschatz, Abraham	11 12	345-399 401-433
Flynn, Ida	7 8	151-178 181-206
Tanenbaum	6	379-399

UNIDAD 6

SEGURIDAD

SUAyED UNAM
Sistema Universidad Abierta y Educación a Distancia

Plan 2012 2016 actualizado

OBJETIVO PARTICULAR

El alumno conocerá los conceptos generales de la seguridad, así como los mecanismos de protección del sistema operativo que permiten salvaguardar los datos que residen en una computadora.

TEMARIO DETALLADO

(8 horas)

6. Seguridad

6.1. Conceptos básicos de seguridad
6.2. Encriptamiento sencillo con llave secreta
6.3. Encriptamiento con llave pública
6.4. Estándares de criptografía
6.5. Capacidades, derechos y matriz de acceso
6.6. Virus y sus variantes
6.7. Contraseñas de una sola vez
6.8. Amenazas, ataques y vigilancia
6.9. Reconstrucción de un sistema violado
6.10. La bitácora o diario de operaciones

INTRODUCCIÓN

El *sistema operativo* puede verse como una extensión de un equipo (*hardware*), ya que a través de él se pueden proporcionar una serie de servicios extras implementados por *software*. También es importante recordar que el sistema operativo controla el acceso a los recursos en un sistema de cómputo y es aquí en donde la seguridad juega un papel importante. La seguridad de los sistemas operativos es sólo una parte del total de la seguridad que debe considerarse en los sistemas de computación. Cualquier tipo de *software* no está libre de fallas y errores y el sistema operativo considerado como un *software* complejo es probable que falle y que afecte a la seguridad del mismo. La única manera de probar la seguridad de un sistema es realizar evaluaciones de seguridad en él. Sin embargo, cuanto más complejo es un sistema, más difícil se vuelve la evaluación de su seguridad. Los sistemas actuales son más complejos, ya que pueden realizar multiproceso, multiusuario, proceso distribuido, cómputo en la nube, virtualización, múltiples aplicaciones, etcétera. En la presente unidad se abordan los aspectos fundamentales que deben tomarse en cuenta para que un sistema operativo funcione de manera segura, considerando que ningún sistema está libre de fallas y que esto implica un proceso continuo de revisión y mejoramiento, ya que cada día surgen nuevos ataques, vulnerabilidades, fallas, etcétera y el sistema operativo no está exento de esto.

6.1. Conceptos básicos de seguridad

Es importante distinguir los términos “seguridad” y “protección”. La seguridad informática implica la protección de uno de los activos más valiosos de las personas y las organizaciones: *la información*, lo que incluye aspectos técnicos, administrativos, legales y políticos. La protección consiste en aplicar los mecanismos específicos del sistema operativo que sirven para salvaguardar la información que comparte una computadora. Sin embargo, la frontera entre los dos conceptos no está bien definida. A continuación, se describen los conceptos generales de seguridad y posteriormente se presentan los mecanismos de protección que ofrecen los sistemas operativos.

- **Sistema de cómputo**. Conjunto formado por:
  - ❑ *Hardware*
  - ❑ *Software*
  - ❑ Datos
  - ❑ Medios de almacenamiento
  - ❑ Recursos humanos involucrados

- **Compromiso**. Se entiende como compromiso de seguridad a cualquier forma posible de pérdida o daño en un sistema de cómputo. De esta forma, comprometer la seguridad de un sistema equivale a la posibilidad de provocar pérdida o daño al sistema.

- **Vulnerabilidad.** Una vulnerabilidad consiste en cualquier debilidad que puede explotarse para causar pérdida o daño al sistema:

- Puede explotarse de manera intencional o bien accidentalmente.
- El punto más débil de seguridad de un sistema, consiste en el punto de mayor vulnerabilidad en ese sistema.

Amenaza. Es cualquier circunstancia con el potencial suficiente para causar pérdida o daño al sistema. Ejemplos:

- Ataques humanos
- Desastres naturales
- Errores humanos inadvertidos
- Fallas internas de *hardware*
- Fallas internas de *software* (sistemas operativos, aplicaciones, etcétera)

Problemas de seguridad

En relación a los problemas de seguridad, Carretero plantea los siguientes puntos (Carretero, 2001: 499-502):

1. Sistemas operativos. “Todos los sistemas operativos comerciales que se han desarrollado, han tenido problemas de seguridad” (Carretero, 2001: 499), sobre todo las computadoras personales y dispositivos portátiles que no fueron diseñados considerando la seguridad. Por ejemplo, las antiguas versiones de Unix en la que cualquier usuario podía leer el archivo “*passwd*”.

2. Uso indebido de *software*. Algunos problemas de seguridad están relacionados con el uso indebido de *software* e inclusive de manera maliciosa. Por ejemplo, se pueden causar fallas de seguridad por medio de las técnicas de “caballo de Troya” para realizar procesos no autorizados y la creación de “puertas traseras” para crear hoyos de seguridad al sistema por medio de asignación de privilegios.

3. Usuarios inexpertos y descuidados. Este tipo de usuarios son muy peligrosos, ya que pueden realizar acciones como: borrar archivos críticos, dejar sesiones abiertas del sistema operativo por demasiado tiempo, o escribir y tener a la vista los *passwords* de acceso. En este caso el administrador del sistema debe estar muy atento y configurar los accesos de ciertos comandos para pedir su confirmación antes de aplicarlos. Por ejemplo, los comandos "*delete*" o "rm", o bien cerrar sesiones de manera automática si estas llevan cierto tiempo abiertas.

4. Usuarios no autorizados. Los sistemas multiusuario como Linux, Unix y Windows NT contienen cuentas de usuario para ingresar al sistema. Estas cuentas deben protegerse por medio de *passwords* que sólo debe conocer el usuario correspondiente y limitar las cuentas a las funciones específicas que debe realizar en el equipo. El proceso de autenticación es muy importante para evitar que usuarios no autorizados accedan al sistema. El usuario no autorizado trata de romper el proceso de autenticación con la identidad de un usuario ilegítimo. Si lo consigue puede realizar lo que desee, desde borrar datos hasta generarse una cuenta verdadera con identidad falsa o cambiar el *password* de la cuenta del administrador para que sólo él pueda acceder al sistema.

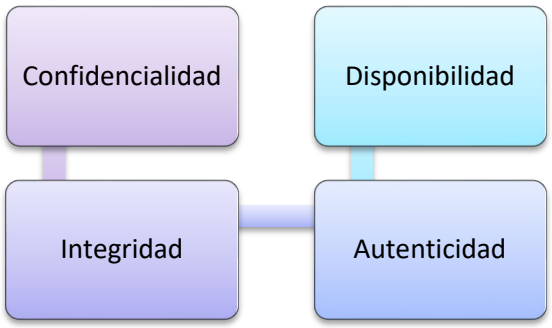
5. Pérdida accidental de datos. Además de los problemas antes mencionados, es posible perder datos valiosos a causa de accidentes y son:
   - Actos fortuitos; incendios, inundaciones, terremotos, guerras e inclusive roedores que dañan los equipos y dispositivos de almacenamiento.
   - Errores de *hardware* o *software*; fallas de CPU, discos, cintas, errores de telecomunicaciones y redes, así como errores de programación.
   - Errores humanos; captura incorrecta de datos, montaje de discos CD-ROM o cintas de manera errónea, ejecución de programas equivocados, extravío o pérdida de un disco o dispositivo de respaldo, etcétera.

Casi todos estos problemas pueden evitarse realizando los respaldos adecuados y de preferencia con los datos originales, en la práctica es probable que la pérdida accidental de datos cause más daño que los intrusos. El tema de respaldos se explicó en la unidad 5.

Seguridad informática

Como todo concepto, la seguridad informática se ha desarrollado y evolucionado dentro de las organizaciones. Con el uso de las tecnologías de información y comunicación la seguridad adquiere nuevas dimensiones en la prevención de delitos y riesgos. La evolución tecnológica genera oportunidades; pero también grandes riesgos a la información, activos, infraestructuras físicas, servidores, aplicaciones, sistemas operativos, etcétera. Por lo que surge la necesidad de reducir los riesgos en el uso de las tecnologías tales como; ataques, fallas, catástrofes, etcétera a través de mecanismos de seguridad, en este caso al tema de los sistemas operativos.

La seguridad informática debe entenderse como un conjunto de estrategias y procesos coordinados para la correcta aplicación de mecanismos que permitan mantener la integridad y disponibilidad de la información centralizada, distribuida y transportada de forma electrónica o física de cualquier organización.



Objetivos de la seguridad informática.

Objetivos de la seguridad informática. Elaboración propia.

Confidencialidad
Que la información sea accesible sólo a aquellos usuarios, entidades y procesos autorizados a tener acceso. ◆ Que la información no sea revelada a entidades no autorizadas. ◆ Aplica a la información durante su almacenamiento, procesamiento o transmisión. **Amenaza: Revelación de datos.**
Integridad
Un sistema posee la propiedad de integridad si los recursos manipulados por éste no son alterados o destruidos por usuarios, entidades y procesos no autorizados. Puede significar cosas distintas dependiendo del contexto: ◆ Precisión. ◆ Exactitud. ◆ Inalterabilidad. ◆ Modificación sólo en modos aceptables. ◆ Modificación sólo por partes o procesos autorizados. ◆ Consistencia. **Amenaza: Alteración de datos.**
Autenticidad
Es que la información provenga de fuentes autorizadas; es una forma de integridad. Ésta puede ser. ◆ **Unilateral**: en la que sólo una parte se autentica ante la otra y la otra parte no se autentica ante la primera. ◆ **Mutua**: ambas partes deben autenticarse entre sí. **Amenaza: Suplantación de identidad.**
Disponibilidad
Que se pueda brindar el acceso a la información y a los recursos relacionados con ella cuando lo requieran los usuarios, entidades y procesos autorizados. Se aplica a datos y recursos. Algunos conceptos asociados a disponibilidad son:

- Presencia de datos y recursos en forma usable.
- Capacidad de responder a necesidades.
- Respuesta en tiempo.

Amenaza: Negación del servicio.

6.2. Encriptamiento sencillo con llave secreta

La palabra criptografía (Tanenbaum, 2003: 724-736) viene del griego y significa "escritura secreta" se remonta a miles de años antes de Jesucristo. Un "cifrado" es una transformación carácter por carácter o *bit* por *bit*, sin importar la estructura lingüística del mensaje, caso contrario al "código" que reemplaza una palabra o símbolo por otro; los códigos actualmente ya no se utilizan. Históricamente varios grupos de personas han utilizado y contribuido al desarrollo de la criptografía, de éstos, la milicia ha tenido el papel más importante y ha moldeado el campo a través de los siglos. La criptografía moderna utiliza las mismas ideas de la criptografía tradicional (la transposición y la sustitución); pero su orientación es distinta. Prácticamente todos los sistemas operativos implementan los diferentes tipos de criptografía para proteger las contraseñas, datos, comunicaciones en red, etcétera.

La criptografía de llave secreta también conocida como "simétrica" se basa en la ocultación de la clave de cifrado, en donde la clave es conocida únicamente por el usuario que cifra el objeto original y utiliza la misma llave para cifrar y descifrar. Estos sistemas ejecutan el cifrado (E) aplicando la clave (k) al objeto origen (O) para obtener el cifrado (C): **C = E (k, O)**.

Características principales:

- El cifrado de clave privada, se conoce también como simétrica o cifrado por bloques.
- Utiliza la misma llave para cifrar y descifrar.
- Se apoya en dos conceptos:

- Confusión. Trata de ocultar la relación que existe entre el texto normal, el texto cifrado y la clave (realiza sustituciones simples).

- Difusión. Trata de repartir la influencia de cada *bit* del mensaje original en el mensaje cifrado (realiza permutaciones).

- Divide el mensaje en bloques de tamaño fijo o variable, esto depende del tipo de algoritmo y aplica la función de cifrado a cada uno de ellos.
- Generalmente es fácil de implementar.
- Pueden implementarse en *hardware* para velocidad de procesamiento o en *software* para flexibilidad en su implementación.
- Los sistemas más comunes de llave privada son: DES, 3DES, IDEA, AES.

6.3. Encriptamiento con llave pública

En un esquema de llave pública todos los usuarios tienen dos tipos de llaves para cifrar y descifrar mensajes; una pública y otra privada. Si alguien quiere enviarte un mensaje, obtiene una copia de tu llave pública con la cual cifra el mensaje que sólo podrá descifrarse con tu llave secreta. Los mensajes cifrados con la llave pública no se pueden descifrar con la misma llave pública. Este tipo de cifrado resuelve varios problemas que tiene el de llave privada, ya que todas las comunicaciones a una entidad utilizan la llave pública de la entidad, reduciendo así el número de llaves que emplea el sistema. Las entidades no se preocupan por mantener secreta su llave, al contrario, la hacen pública y se reduce el riesgo de que una llave privada caiga en manos de una persona malintencionada, ya que cada usuario procura mantener secreta su llave descifradora, que es su llave privada. El cifrado de llave pública está basado en funciones matemáticas complejas (logaritmos discretos y curvas elípticas) que hace poco posible que, con un tiempo y capacidad de cómputo razonable, conociendo sólo el mensaje cifrado y la llave pública pueda obtenerse la llave privada y con ella obtener el mensaje original.

El proceso de cifrado es el siguiente:

- Cada usuario, genera un par de llaves que serán utilizadas para cifrar y descifrar los mensajes que se van a recibir.
- Si Alicia le quiere enviar un mensaje a Roberto lo cifra usando la llave pública de Roberto.
- Cuando Roberto recibe el mensaje, lo descifra con su llave privada.

Características principales:

- El cifrado de llave pública es también conocido como criptografía asimétrica.
- Utiliza dos llaves; pública y privada (para cifrar y descifrar).
- El emisor y receptor deben tener el par de llaves acopladas.
- Una de las dos llaves debe conservarse en secreto.
- Emplea longitudes de clave mayores que los de llave secreta.
- La complejidad de cálculo los hace más lentos que los de llave privada.
- Es más segura que los de llave privada.

6.4. Estándares de criptografía

La criptografía se utiliza para intercambiar mensajes que sólo pueden ser leídos por usuarios a los que van dirigidos y que poseen los medios para descifrarlos. Como hemos visto, existen dos tipos de estándares de criptografía que se clasifican por el número de llaves que utilizan (llave secreta y llave pública), cada una de éstas utiliza estándares criptográficos para implementar el cifrado de manera práctica, a continuación, se describen los más comunes:

Estándares de llave secreta

1. DES (*Data Encription Estandar*). Es el sistema más popular de la criptografía de llave secreta. Se encarga de realizar combinaciones, sustituciones y permutaciones entre el texto a cifrar y la clave, asegurando de manera simultánea que las operaciones puedan realizarse en ambas direcciones (descifrado); a la combinación entre sustituciones y permutaciones se le conoce como cifrado del producto. DES

cifra bloques de 64 *bits*, mediante llaves de 56 *bits*, esto hace que existan 256 = 7.2 X 10_{16} llaves diferentes.

Ventajas:
- Es uno de los sistemas más empleados y extendidos.
- Sistema muy probado.
- Implementación sencilla y rápida.

Desventajas:
- No se permite una clave de longitud variable para tener mayor seguridad.
- Es vulnerable al permitir sólo una longitud de clave de 56 *bits*.
- Existen sistemas más robustos.

2. 3 DES (*Triple Data Encryption Standard*). Es una variante de DES. Se basa en aplicar el algoritmo DES tres veces con diferentes claves al mensaje original. Su estructura es la siguiente: Codifica con una subclave k1, decodifica con k2 y vuelve a codificar con k1. La clave resultante es la concatenación de k1 y k2 con una longitud de 112 *bits*.

Ventajas:
- Aumenta de forma significativa la seguridad de DES.

Desventajas:
- Requiere más recursos de cómputo para cifrar y descifrar.

3. IDEA (*International Data Encryption Algorithm*). Codifica bloques de 64 *bits* empleando una clave de 128 *bits*.

Ventajas:
- Algoritmo bastante seguro, resistente a los ataques.

- Su longitud de clave hace imposible un ataque de fuerza bruta.
- Muy utilizado.

Desventajas:
- Requiere mayor procesamiento y velocidad.

Estándares de llave pública

1. RSA (Rivest-Shamir-Adelman). Es el estándar más común en sistemas de llave pública. RSA es seguro si se utilizan llaves suficientemente largas (512 *bits* es poco seguro, 768 *bits* es moderadamente seguro y 1024 *bits* es muy seguro).

Ventajas:

- Es uno de los estándares de llave pública más seguros.
- RSA se basa en la dificultad para factorizar grandes números.
- El atacante se encontrará con un problema de factorización.
- Se utiliza para la autenticación de mensajes y para el intercambio de llaves secretas.

Desventajas:

- Es vulnerable si las claves son demasiado cortas.
- RSA es vulnerable a ataques en texto claro conocido.
- Es un muy lento.

2. Diffie-Hellman. Es un esquema para realizar la generación secreta de llaves de sesión, que permite a dos usuarios acordar una clave común a través de un canal de comunicación inseguro, para que posteriormente puedan usarse para cifrar mensajes.

Ventajas:

- No son necesarias las claves públicas en el sentido estricto; sino una información compartida por los dos comunicantes.
- Su efectividad depende de la dificultad de calcular logaritmos discretos.

Desventajas:

- El estándar está limitado al intercambio de llaves.

1. DES (*Data Encription Estandar*). Es el sistema más popular de la criptografía de llave secreta. Se encarga de realizar combinaciones, sustituciones y permutaciones entre el texto a cifrar y la clave, asegurando de manera simultánea que las operaciones puedan realizarse en ambas direcciones (descifrado), a la combinación entre sustituciones y permutaciones se le conoce como cifrado del producto. DES cifra bloques de 64 *bits*, mediante llaves de 56 *bits*, esto hace que existan 256 = 7.2 X 10_{16} llaves diferentes.

Ventajas:

- Es uno de los sistemas más empleados y extendidos.
- Sistema muy probado.
- Implementación sencilla y rápida.

Desventajas:

- No permite una llave de longitud variable para ofrecer mayor seguridad.
- Es vulnerable ya que permite una llave de 56 *bits*.
- Actualmente existen sistemas más robustos.

2. 3DES (*Triple Data Encryption Standard*). Es una variante de DES Se basa en aplicar el algoritmo DES tres veces con diferentes claves al mensaje original. Su estructura es la siguiente: Codifica con una subclave k1, decodifica con k2 y vuelve

a codificar con k1. La clave resultante es la concatenación de k1 y k2 con una longitud de 112 *bits*.

Ventajas:
- Aumenta de forma significativa la seguridad de DES.

Desventajas:
- Requiere más recursos de cómputo para cifrar y descifrar.

3. IDEA (*International Data Encryption Algorithm*) Codifica bloques de 64 *bits* empleando una clave de 128 *bits*.

Ventajas:
- Algoritmo bastante seguro, resistente a los ataques.
- Su longitud de clave hace imposible un ataque de fuerza bruta.
- Muy utilizado.

Desventajas:
- Requiere mayor procesamiento y velocidad.

4. Firmas digitales. Una firma digital es un mecanismo de seguridad criptográfica que asocia una identidad de una persona, equipo, documento o archivo a un mensaje transmitido de manera electrónica. La firma digital, consiste en dos procesos: el de firma y el de verificación de firma.

Sello y Huella.

El proceso de combinar la firma digital con la criptografía de llave pública da por resultado la seguridad del cifrado con la autenticidad de la firma digital.

Ventajas:

- Permite firmar mensajes de correo electrónico y otros documentos digitales, de tal forma que no puede ser negado por quien los envió.

6.5. Capacidades, derechos y matriz de acceso

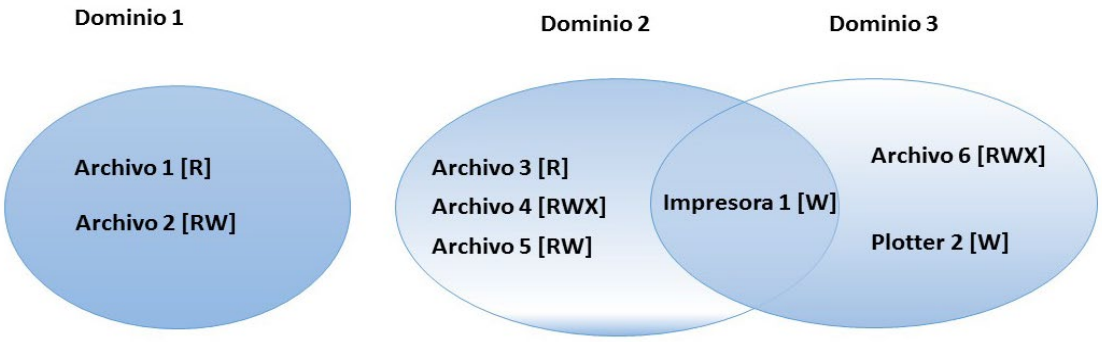
Una de las funciones principales del sistema operativo, es proteger los recursos de los usuarios para que puedan trabajar en un entorno de forma segura. El sistema operativo tiene que asegurar que el uso de los recursos del sistema (uso de procesador, memoria, disco, *software*, archivos, bases de datos y otros recursos), sólo sean operados por los usuarios y procesos autorizados por el propio sistema. La protección se realiza por medio de mecanismos para hacer cumplir las políticas que rigen el uso de los recursos. Los mecanismos determinan **cómo se realizará algo** y las políticas deciden **qué se hará.** Por ejemplo, el sistema Unix proporciona el mecanismo de protección para establecer en los archivos el valor de los *bits* (*read, write, execute*) para el propietario, grupo y universo. La política para establecer el valor de estos *bits* se deja a los usuarios o al administrador del sistema operativo.

De acuerdo con Silberschatz, (2002: 567-569), existen las siguientes metas de la protección:

- Impedir la violación maliciosa e intencional por parte de un usuario sobre una restricción definida previamente por el administrador o el sistema operativo.

- Asegurar que cada *software* o programa utilice los recursos de acuerdo a las políticas establecidas.
- Detectar errores latentes en las interfaces de los subsistemas para impedir la contaminación de un subsistema que está funcionando correctamente.

Para comprender mejor los diferentes mecanismos de protección, es importante conocer el concepto de dominio de protección. Según lo plantea Tanenbaum, “Un dominio de protección es un conjunto de pares (objeto, derechos). Cada par especifica un objeto y algún subconjunto de las operaciones que pueden ejecutarse con él. Un derecho implica el permiso para realizar una de las operaciones” (Tanenbaum, 2003: 645). Un dominio puede corresponder a un solo usuario para indicar lo que puede y no puede realizar, aunque un dominio puede ser más general que un solo usuario. El siguiente esquema muestra un ejemplo de los dominios y el conjunto de permisos sobre los objetos.



Elaboración con base en: http://bit.ly/1SHtUWb

Capacidades.

Una de las formas para implementar la protección, es asociar a los dominios las descripciones (capacidades) de las operaciones que los componentes de un dominio pueden realizar sobre cada objeto del sistema. Las capacidades (*capabilities*) son el resultado de la combinación de alguna referencia a un objeto con los permisos de acceso al mismo desde el dominio poseedor de la capacidad. En la siguiente tabla se muestra un ejemplo general los permisos asignados que cada dominio tiene respecto al objeto (matriz de derechos).

Objeto/ Dominio	F1	F2	F3	Lector de tarjeta	Impresora
D1	Lectura		Lectura		
D2				Lectura	Impresión
D3		Lectura	Ejecución		
D4	lectura/Escritura		lectura/Escritura		

Elaboración con base en: http://1.bp.blogspot.com/-5INwDsGP56A/ThvjuNMu24I/AAAAAAAAABA/X4vOcTIQ_iY/s1600/sp.jpg Fecha: 19/05/2016

Lista de capacidades

De acuerdo con Carretero (2001: 538) cuando se utiliza el método de capacidades, cada proceso tiene asociado una lista de objetos a los que pueda acceder y la indicación de las operaciones que puede realizar sobre los objetos (dominio). Esta lista, se denomina ***lista de capacidades.*** Las listas de capacidades son a su vez objetos y pueden ser incluidas dentro de otras listas para facilitar la compartición de objetos en dominios y subdominios. Se recomienda revisar los ejemplos descritos en la unidad 5.3.

Derechos 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.[24]

Matriz de acceso

Según los planteamientos de Carretero: "La relación entre dominios y objetos se puede definir de forma completa mediante una matriz de protección, también denominada de acceso" (2001: 534). La matriz está compuesta por filas y columnas, en donde las filas corresponden a los dominios y las columnas a los objetos. Cada celda indica los derechos que el dominio tiene sobre los objetos, si es que los tiene, las celdas vacías indican que no existen derechos. Por ejemplo, la siguiente figura muestra los elementos (D, F) que expresa las operaciones que los dominios "D" puede realizar sobre los objeto "F". Si la matriz de protección está definida de forma completa, los mecanismos de protección siempre sabrán qué hacer cuando un proceso de un dominio solicita cierta operación sobre el objeto.

Objeto/Dominio	F1	F2	F3	Impresora
	Leer		Leer	
D1				
D2				Imprimir
D3		Leer	Ejecutar	
D4	Leer - escribir		Leer-escribir	

Elaboración con base en: Silberschatz: 574.

También es posible que se realice la conmutación de dominio que consiste en una combinación (dominio, objeto) y sus derechos de acceso y en la que los dominios son un objeto más del sistema, por lo cual se incluyen los cambios de

[24] Salvador Meza Badillo, *Sistemas Operativos Multiusuario*, SUA-FCA-UNAM, pp. 43-4, material disponible en línea: http://fcasua.contad.unam.mx/apuntes/interiores/docs/98/2/sis_operativos.pdf, recuperado el 09/04/2013.

dominio permitidos. La siguiente figura muestra la matriz con cuatro dominios como objetos, los procesos del dominio D1 pueden cambiar el dominio D4, pero una vez realizados no podrán regresar.

La matriz de protección tiene algunos inconvenientes para su implementación:

- La matriz de un sistema complejo puede ser demasiada grande y dispersa.
- La matriz tiene un número fijo de dominios y objetos, por lo que es poco flexible para sistemas en la que el número de dominios y objetos pueden cambiar.

Para resolver esta problemática, la mayoría de los sistemas operativos utilizan la matriz de estructuras dinámicas (listas) a las que se les puede agregar o eliminar elementos sin tener que redefinir alguna estructura de datos del sistema. También se evita la dispersión de la matriz por medio de elementos dinámicos. Para lograr esto, existen dos enfoques: **Listas de control de acceso** (*ACL, Access Control, List*) y **por capacidades** (*capabilities*), descritos en esta unidad en el apartado 5.3.

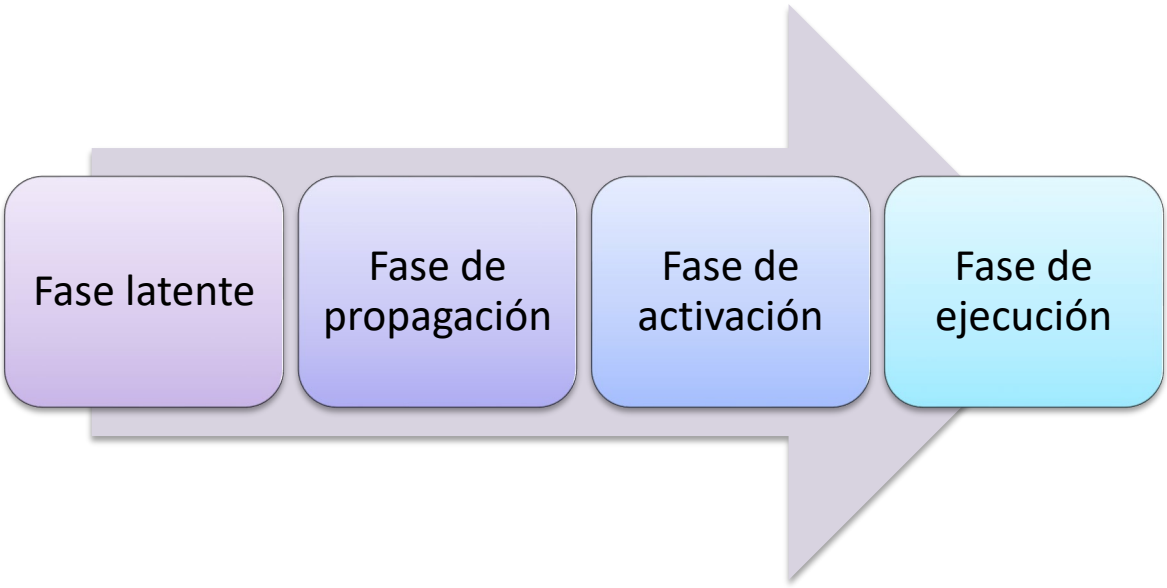
6.6. Virus y sus variantes

De acuerdo con Carretero (2001: 502-505), se denomina virus a un programa que se auto replica con fines destructivos o de violación de seguridad. Se llaman así por la analogía con los virus que actúan sobre los seres vivos. Los virus requieren de un programa que transporte el virus y de un agente que los transmita para infectar a otros programas. En todos los casos se trata de un programa o parte de código que se añade a otro y que se activa cuando se ejecuta el programa portador; es decir, se ejecuta el portador y el virus.

El virus, una vez que se ha alojado en una computadora, toma el control temporal del sistema operativo y, ya infectado, entra en contacto con un elemento del *software* no infectado, se pasa una nueva copia del virus al programa. Así pues, la infección puede extenderse de un equipo a otro, a través de usuarios no sospechosos que intercambian sus discos o dispositivos portátiles, o bien se envían programas de uno a otro a través de la red. En un entorno de red, la capacidad de acceder a las aplicaciones y los servicios del sistema de otro equipo hace que la propagación de virus afecte no sólo al equipo inicialmente infectado; sino que afecta a todos los equipos de una red de computadoras.

Un virus puede hacer cualquier cosa que hagan otros programas. La única diferencia es que se acopla a otro programa y se ejecuta de forma oculta cada vez que se ejecuta el programa anfitrión. Una vez que un virus se está ejecutando, puede realizar cualquier función, como borrar archivos y programas.

Las etapas o fases de un virus son las siguientes:



Elaboración propia.

1. *Fase latente.*

   El virus está inactivo. El virus se activará por algún suceso, como una fecha, la presencia de otro programa o archivo o que la capacidad del disco exceda cierto límite; es importante comentar que no todos los virus pasan por esta etapa.

2. *Fase de propagación.*

   El virus hace copias idénticas de él en otros programas o en ciertas áreas del sistema del disco. Cada programa infectado contendrá ahora un clon del virus, que entrará a su vez en la fase de propagación.

3. *Fase de activación.*

   El virus se activa para llevar a cabo la función para la que está diseñado. Como en la fase latente, la fase de activación puede producirse por múltiples sucesos del sistema, incluyendo una cuenta del número de veces que esta copia del virus ha hecho de sí mismo.

4. Fase de ejecución.

Se lleva a cabo la función. La función puede ser no dañina, como dar un mensaje por la pantalla, o dañina, como la destrucción de archivos de programas y datos.[25]

La mayoría de los virus llevan a cabo su trabajo de manera específica para un sistema operativo concreto y, en algunos casos, para una plataforma en particular. Así pues, están diseñados para obtener ventaja de los detalles y las debilidades de un sistema en concreto.

Tipos de virus.

Desde que los virus aparecieron, se ha producido una carrera entre los escritores de virus y los escritores de *software* antivirus. A medida que se han desarrollado contramedidas eficaces para los tipos de virus existentes, se han desarrollado nuevos tipos de virus.

Variantes de virus

Virus parásitos	La forma más tradicional y común de virus. Un virus parásito se agrega a archivos ejecutables y se reproduce al ejecutar el programa infectado, buscando otros archivos ejecutables qué infectar.
Virus residentes en la memoria	Se alojan en la memoria principal como parte de un programa del sistema residente. A partir de esto, el virus infecta todos los programas que se ejecutan.
Virus del sector de arranque	Infecta el sector principal de arranque (*master boot record*) o sector de arranque y se propaga cuando el sistema arranca desde el disco que contiene el virus.

[25] Departamento de Informática, Universidad Nacional del Nordeste, Argentina, *Sistemas operativos, Seguridad,* material en línea, disponible en: http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/MonogSO/SEG02.html Fecha de recuperación: 09 de enero de 2009.

Virus clandestino	Una forma de virus diseñado explícitamente para esconderse de la detección mediante un *software* antivirus.
Virus polimorfo	Un virus que muta con cada infección, haciendo imposible la detección por la «firma» del virus.

Un ejemplo de **virus clandestino** es el que utiliza compresión para que el programa infectado tenga exactamente la misma longitud que una versión no infectada. Existen técnicas más sofisticadas. Por ejemplo, un virus puede poner alguna lógica de interceptación en las rutinas de E/S a disco, de modo que cuando haya un intento de leer partes sospechosas del disco mediante estas rutinas, el virus presente el programa original no infectado. Así pues, el término clandestino no se aplica a los virus como tales; sino que, más bien, es una técnica empleada por los virus para evitar su detección.

Así mismo, un virus **polimorfo** crea copias durante la reproducción que son funcionalmente equivalentes, pero que tienen diferentes patrones de "*bits*".
Como con los virus clandestinos, su finalidad es vencer al *software* antivirus, la "firma" que generan varía con cada copia, esto lo logra, insertando de forma aleatoria instrucciones muy cortas o intercambiando el orden de las mismas e inclusive utilizado técnicas criptográficas.

"Otra herramienta de los programadores de virus son un juego de utilerías para la creación de virus."[26] Dicho juego, permite que una persona con pocos conocimientos técnicos cree rápidamente una serie de virus diferentes. Aunque los creados con estas utilerías tienden a ser menos sofisticados que los diseñados desde cero, el número absoluto de nuevos virus que pueden generarse crea un problema para los procedimientos y productos antivirus.

[26] Departamento de Informática, Universidad Nacional del Nordeste, Argentina, *Sistemas operativos, Seguridad,* material en línea, disponible en: http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/MonogSO/SEG02.html Fecha de recuperación: 09 de enero de 2009.

Otra herramienta más del programador de virus es anunciar por internet su creación y sus efectos, el cual utiliza este medio para el intercambio de los mismos. Varios anuncios de este tipo han surgido en los Estados Unidos y en otros países. Estos anuncios ofrecen copias de virus que pueden obtenerse por la red, así como consejos para su creación.

Malware más común

- **Gusanos.** Los gusanos o *worms* son programas independientes con capacidad de replicarse a sí mismos para afectar a más sistemas. Generalmente, se desarrollan para acceder por medio de redes locales y remotas para explotar fallas de seguridad de los sistemas operativos y protocolos de comunicación. Su capacidad es tal, que pueden provocar la caída de miles de computadoras antes de ser detectados y eliminados. El principal problema de los gusanos no es que sean destructivos, sino que colapsan las redes de comunicaciones provocando los ataques conocidos como negación del servicio (DoS) y negación de servicio distribuido (DDoS).

- **Caballos de Troya.** Son programas con apariencia inocente que contienen código para realizar una función inesperada e indeseable. Su ejecución puede consistir en borrar, cifrar, modificar, copiar archivos en otro lugar para ser utilizados posteriormente por medio de *e-mail*, instalar virus, etcétera. Para lograr su ejecución, el usuario del equipo ejecuta un programa que lo instala, generalmente por medio de programas bajados de internet; juegos, *software* nuevo y gratuito, pornografía, o anuncios vistosos que invitan a probar ciertos programas.

- **Bombas lógicas.** Son un fragmento de código escrito e insertado en el sistema operativo en función. Se dispara cuando se cumplen ciertas condiciones como; fechas específicas, inicio del sistema, secuencia definida de teclas, etcétera. Su efecto puede provocar borrado del disco, borrado de archivos, cambios difíciles de detectar a programas importantes, cifrar archivos, etcétera. También puede utilizarse para enviar virus y gusanos.

6.7. Contraseñas de una sola vez

La seguridad de las contraseñas es un elemento muy importante en la protección de los sistemas operativos. Existen diversos mecanismos para proteger el acceso al sistema y una de ellas es establecer la política de cambiar la contraseña para usarse sólo una vez (*One-Time Password, OTP*). Esta variante de autenticación dificulta el acceso no autorizado, ya que la contraseña utilizada sólo es válida para una sesión. Cuando se usa este tipo de caracteres clave, se entrega al usuario un libro que contiene una lista de las mismas. En cada inicio de sesión, se usa la siguiente contraseña de la lista. Si un intruso llega a descubrirla, no le servirá de nada; pues en la siguiente ocasión deberá usarse una distinta. Es importante para el usuario no perder el libro con estas claves de acceso.

En realidad, no se requiere del libro de contraseñas, ya que existe un esquema ideado por Leslie Lamport (Computólogo estadounidense) que utiliza algoritmos matemáticos para generar las contraseñas en una computadora que sea segura. Si la comunicación es insegura, cuando se utilizan los protocolos (*ftp, telnet*) se garantiza que aunque se descubran las contraseñas, éstas no se podrán utilizar. Si la comunicación es segura utilizando el protocolo "*ssh*" con contraseña, se evita la posible ruptura de la protección o lectura de teclado (*keyloggers*). El método de

Lamport puede servir para que un usuario inicie sesión en un servidor a través de internet desde su computadora personal, aunque los intrusos puedan ver y copiar todo el tráfico en ambas direcciones. Además, no es necesario almacenar secretos en el sistema de archivos del servidor ni en el de la computadora del usuario.

El autor Tanenbaum (2003: 599-600) explica cómo se generan las claves de una sola vez de la siguiente forma: "El algoritmo se basa en una función unidireccional; es decir, una función $y = f(x)$ con la propiedad de que, dado x, es fácil calcular y,; pero dado y no es factible determinar x desde el punto de vista computacional. La entrada y la salida deben tener la misma longitud, por ejemplo, 128 *bits*.

El usuario escoge una contraseña secreta que memoriza, también escoge un entero, n, que es el número de contraseñas para usarse sólo una vez y que el algoritmo podrá generar. Por ejemplo, consideremos $n = 4$, aunque en la práctica se usaría un valor mucho más grande. Si la contraseña secreta es S, la primera contraseña se obtendrá ejecutando la función unidireccional *n* veces:

$$P_1 = f(f(f(f(s))))$$

La segunda contraseña se obtiene ejecutando la función unidireccional $n-1$ veces:

$$P_2 = f(f(f(s)))$$

La tercera contraseña ejecuta f dos veces, y la cuarta, una vez. En general $P_{i-1} = f(P_i)$. Lo más importante por observar aquí es que, dada cualquier contraseña de la secuencia, es fácil calcular la anterior en orden numérico; pero

imposible calcular la siguiente. Por ejemplo, dada P_2 es fácil determinar P_1; pero imposible determinar P_3.

El servidor se inicia con P_0, que no es sino $f(P_1)$. Este valor se almacena en el archivo de contraseñas, en la entrada correspondiente al nombre de inicio de sesión del usuario, junto con el entero 1, lo que indica que la siguiente contraseña requerida será P_1. Cuando el usuario quiere iniciar sesión por primera vez, envía su nombre de inicio de sesión al servidor, el cual responde enviando el entero que está en el archivo de contraseñas 1 La máquina del usuario responde con P_1, que puede calcularse en forma local a partir de *s*, la cual se teclea en el momento. Entonces el servidor calcula $f(P_1)$ y compara esto con H, valor almacenado en el archivo de contraseñas (P_0). Si los valores coinciden, se permite iniciar sesión, el entero se incrementa a 2, y P_1 sobrescribe a P_0 en el archivo de contraseñas.

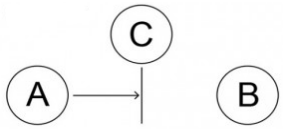
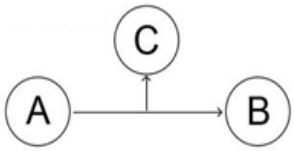
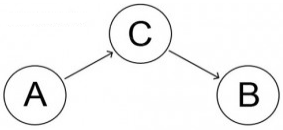
En el siguiente inicio de sesión, el servidor envía un 2 al usuario, y la máquina de éste calcula P_2. Luego el servidor calcula $f(P_2)$ y lo compara con la entrada del archivo de contraseñas. Si los valores coinciden, se permite iniciar la sesión, el entero se incrementa a 3 y P_2 sobrescribe a P_1 en el archivo de contraseñas. La propiedad que hace que funcione este esquema es que, aunque un *cracker capture* P_1, no tiene forma de calcular P_{i+1} a partir de ella; sólo podrá calcular P_{i-1} que ya se usó y ha quedado inutilizada". Cuando se hayan utilizado todas las contraseñas, el servidor volverá a iniciarse con una nueva clave secreta.

6.8. Amenazas, ataques y vigilancia

Tipos de amenazas

Los principales activos o recursos que hay que proteger en un sistema de cómputo son: *hardware*, *software* y datos.

Existen cuatro tipos de **amenazas** principales a los sistemas que explotan las vulnerabilidades de los activos en el sistema y son: interrupción, intercepción, modificación y fabricación.

◆ **Interrupción** 	El activo de un sistema se pierde o se hace no disponible o inutilizable. Ejemplo: la destrucción maliciosa de un dispositivo, borrado de un programa o archivo, borrado total o parcial de un sistema operativo.
◆ **Intercepción** 	Un sujeto no autorizado logra el acceso a un activo del sistema (problemas de confidencialidad). Ejemplo: la intervención de un canal de comunicación para obtener datos sobre la red.
◆ **Modificación** 	Una entidad no autorizada logra el acceso al activo y puede manipular el servicio que proporciona (problemas de integridad). Ejemplo: modificar los datos en una comunicación.
◆ **Fabricación**	Una entidad no autorizada puede fabricar objetos falsos en un sistema (problemas con la autenticidad). Ejemplo: inserción de transacciones falsas en un sistema de comunicación en red.

Ataques

De acuerdo a lo anterior, un ataque consiste en cualquier acción que explota una vulnerabilidad. Los ataques pueden ejecutarse por diversos motivos:

- Para obtener acceso al sistema.
- Para robar información, como secretos industriales o propiedad intelectual.
- Para recopilar información personal acerca de un usuario.
- Para obtener información de cuentas bancarias.
- Para obtener información acerca de una organización (la compañía del usuario, etcétera).
- Para afectar el funcionamiento normal de un servicio.
- Para utilizar el sistema de un usuario para un ataque a otros sistemas.
- Para usar los recursos del sistema del usuario, en particular cuando la red en la que está ubicado tiene un ancho de banda considerable.

Los ataques se pueden clasificar en pasivos y activos.

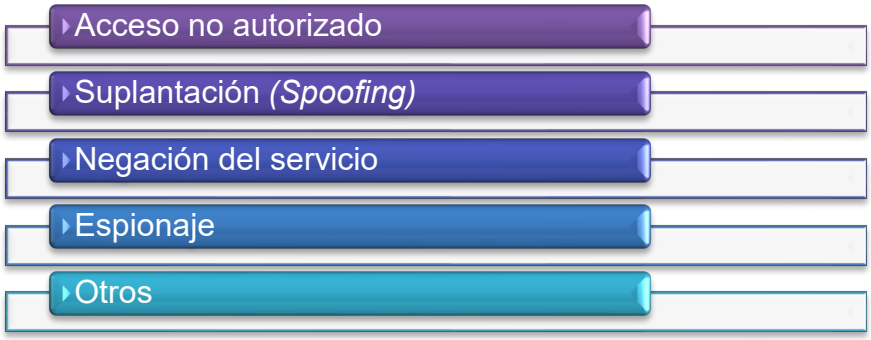
Un ataque pasivo consiste en sólo observar comportamientos o leer información, sin alterar el estado del sistema ni la información. Un ataque pasivo sólo afecta la confidencialidad o privacidad del sistema o de la información.

Un ataque activo tiene la capacidad de modificar o afectar la información, el estado del sistema o ambos. Normalmente un ataque pasivo es la antesala o preparación para un ataque activo.

El atacante es capaz de: interceptar, leer, alterar, modificar, cambiar, fabricar, retener o reenviar información. Así como también puede engañar y suplantar a las partes legítimas en una comunicación.

En todos los casos, el objetivo de un atacante siempre es aprovechar en su favor la información, el medio, o los recursos de cómputo. Para ello, se asume que el atacante conoce, o puede conocer, la naturaleza y estructura de la información, los algoritmos de cifrado (si se usan) y, sobre todo, la implementación de ellos.

La mayoría de los ataques se pueden englobar de la siguiente forma:



El **acceso no autorizado** se da cuando una entidad no autorizada logra acceder a un activo y tiene la posibilidad de alterarlo. El acceso se da por la intercepción de información que se transmite sobre un canal de comunicación inseguro o la explotación de una vulnerabilidad inherente a la tecnología o producto. Cualquier atacante que se dedica a registrar la información que se transmite en la red (ataque pasivo) puede posteriormente, inferir información a través del examen de atributos y realizar un daño a los activos (ataque activo). Ejemplos:

- Un sistema operativo mal configurado y sin actualizaciones de *software*.
- Un servidor de datos, sin protección en la red local (LAN).
- *Password* débil de la cuenta de *root* o de administración del equipo.

La **negación del servicio (DoS)** consiste en la interrupción del servicio, debido a la destrucción o inhabilitación del uso de los recursos de la red. Esto es que un atacante sobrecarga un servicio de red, no afectando la información, sólo

obstaculiza su acceso. Constituye una de las principales amenazas a la seguridad en red. Los tipos de negación del servicio son:

- Saturar de peticiones al sistema operativo de un equipo.
- Inundar la red con tráfico que afecte a varios equipos o la red completa.
- Bloqueo de transmisiones basado en direcciones lógicas.
- Réplica de mensajes de manera continua.

La negación del servicio también puede darse por realizar daños a la infraestructura física de la red o de algún equipo (robo de un disco, daño eléctrico, robo de un dispositivo de la red, etcétera).

La **suplantación** es la capacidad que tiene un atacante para registrar, retener, modificar, sustituir, y reenviar información, comúnmente conocido como "*spoofing*". Constituye una de las más grandes amenazas a la seguridad. Este tipo de ataques se realizan a través de identidades falsas para obtener accesos no autorizados o modificar información con fines ilícitos. Ejemplo, una cuenta no autorizada con los permisos de administrador o *root*, puede alterar y dañar todo el sistema operativo, sus aplicaciones, usuarios, información, etcétera.

El **espionaje** consiste en capturar información mientras se transmite de un punto a otro (origen-destino), para realizar este tipo de ataques se requiere de un equipo especial y un conocimiento avanzado del funcionamiento de los protocolos de comunicación. Los ataques de este tipo generalmente son:

- Intercepción de señales de radiofrecuencia.
- Intervención de canales de comunicación.
- Emisión de señales desde equipos de comunicaciones.
- Captura de paquetes *Ethernet* que no están protegidos.

Lo anterior da como resultado que se pueda obtener los *password* de los equipos y los usuarios para causar daño.

Otros tipos de ataques

Ingeniería social. En muchos casos, el usuario es quien, por ignorancia o por causa de un engaño, genera una vulnerabilidad en el sistema al proporcionar información (contraseña, abrir un archivo, etcétera) al atacante. Cuando esto sucede, ningún dispositivo puede proteger al usuario contra la falsificación: sólo el sentido común, la razón y el conocimiento básico acerca de las prácticas utilizadas pueden ayudar a evitar este tipo de errores.

Puertas trampa. Son puertas traseras ocultas en un programa de *software* que brindan acceso a su diseñador en todo momento.

Es por ello que los errores de programación de *software* son corregidos con bastante rapidez por su diseñador cuando se publica la vulnerabilidad. En consecuencia, los administradores o usuarios siempre deben mantenerse informados acerca de las actualizaciones de los programas del sistema, de aplicación, etcétera, con el fin de limitar los riesgos de ataques.

Existen dispositivos (*firewalls*, sistemas de detección de intrusiones, antivirus) que brindan la posibilidad de aumentar el nivel de seguridad.

Vigilancia

De acuerdo con Silberschatz, (2002: 603-604), la seguridad de los sistemas operativos puede mejorarse a través de dos técnicas administrativas; el monitoreo o vigilancia de amenazas y la bitácora de auditoría. El monitoreo consiste en que el sistema pueda verificar si existen patrones sospechosos en la actividad que el equipo realiza y pueda detectar un intento de violación de la seguridad. Por ejemplo,

cuando un equipo multiusuario y de tiempo compartido cuenta los intentos realizados de un usuario que quiere conectarse o el número de contraseñas ingresadas de forma incorrecta. La bitácora de auditoría registra la hora, usuario y los tipos de acceso a un objeto. Cuando la seguridad ha sido violada, los administradores deben utilizar esta técnica para determinar cómo y cuándo ocurrió el problema, así como determinar el daño causado. La información obtenida es útil para recuperar un equipo y permite desarrollar medidas de seguridad más estrictas para evitar problemas posteriores. El tema de las bitácoras se desarrolla de manera más amplia en el tema 6.10.

Aunque el uso de las bitácoras es importante, se puede explorar periódicamente el sistema en busca de fallas o agujeros de seguridad. La exploración del sistema puede realizarse cuando el equipo tenga poca actividad de procesamiento y uso de la red, por lo que será más práctico que el uso de bitácoras.

La monitorización (vigilancia) puede verificar lo siguiente:

- Contraseñas cortas y sencillas de adivinar.
- Programas "*setuid*" (permisos de acceso para Unix) no autorizados.
- Instalación de programas no autorizados en directorios.
- Procesos que consumen mucho tiempo.
- Protección de directorios del sistema y de usuarios de forma incorrecta.
- Protección de archivos del sistema, dispositivos y *kernel* de forma incorrecta. Por ejemplo. El archivo de contraseñas.
- Acceso peligroso en la ruta de búsqueda de programas. Por ejemplo, caballos de Troya.
- Cambios a programas del sistema detectados con valores de suma de verificación alterados.
- Otros.

Cualquier problema encontrado puede corregirse de forma automática o ser reportado a los responsables del sistema. Como se mencionó anteriormente, los equipos conectados en red son más susceptibles de sufrir ataques a su seguridad que los equipos independientes, ya sea por la debilidad de algunos protocolos de comunicación o por dispositivos de conexión como los *modems* y *switches*, por lo que se hace necesario realizar un monitoreo completo y continuo de toda la infraestructura de comunicaciones instalada y también tomar acciones de protección. Por ejemplo, instalación de *firewalls*, detectores de intrusos, segmentación de redes, etcétera.

6.9. Reconstrucción de un sistema violado

Cualquier suceso que afecte el funcionamiento normal de una computadora se considera como un desastre. Esto puede incluir la destrucción del registro de inicio que se almacena en un dispositivo del sistema, borrado de uno o más archivos del sistema operativo, la destrucción de un dispositivo físico, etcétera. Lo anterior puede generarse por acceder de forma no autorizada al sistema o también por fallas humanas que afectan la fiabilidad y la disponibilidad del sistema. Una vez que el administrador del sistema tenga sospechas que el sistema operativo ha sido violado, deberá realizar pruebas de integridad del sistema para confirmar que el sistema realmente fue violado. Las pruebas pueden realizarse con utilerías o comandos del sistema operativo o por *software* desarrollado por terceros, por ejemplo: "*Tripwire*" o AIDE (*Advanced Intrusion Detection Environment*) para Linux o Windows. Los tipos de pruebas que deberá realizar el administrador del sistema son principalmente: verificar los registros del procesador, estructura de datos en la memoria *swap*, estructuras de datos de red, contadores, procesos, de usuario en memoria y *stacks*, *caché* del *file system, file system*, etcétera.

Reconstrucción

La reconstrucción o recuperación ante un desastre es la reconstrucción de una computadora de modo que se pueda iniciar la sesión y acceder a los recursos del sistema después de ocurrir el desastre del sistema. Las principales acciones que tiene que realizar el administrador son las siguientes:

- Iniciar el sistema operativo en modo a prueba de errores.
- Utilizar utilerías del sistema operativo (*Recovery Console*)
- Utilizar discos de reparación de emergencia (generados en la instalación del sistema operativo).
- Cambiar *password* de "*root*" o administrador
- Verificar la integridad del sistema operativo.

Una vez restaurado el sistema operativo, se debe de proceder a la restauración de los datos por medio de:

- Realizar el proceso de *restore* de los *backups* (previamente realizados)
- Seleccionar los archivos/directorios a restaurar.
- Verificar la seguridad del sistema de archivos.
- Documentar a detalle el incidente.
- Iniciar el servicio a los recursos del sistema. (Microsoft, 2001: 658-668).

6.10. La bitácora o diario de operaciones

Las bitácoras o *logs* son archivos y directorios en los que se registran las actividades que se realizan en un sistema durante un tiempo en particular. Ante una falla o suceso relevante de un dispositivo o sistema es necesario conocer qué sucedió y es cuando surge la pregunta sobre la existencia de bitácoras y, si existen, sobre quién las administra e incluso por cuánto tiempo se guardan. También son utilizadas como pistas de auditoría para rastrear incidentes o eventos que pudieron afectar o dañar un proceso, programa, sistema, etcétera.

Las bitácoras registran datos o información sobre:

- ¿Qué? Nos permite responder cuál fue el elemento manipulado.

- ¿Quién? Apunta hacia la entidad que interactuó con el elemento de nuestro interés. Puede ser una persona, un sistema, un grupo o una dirección IP. Esta información proporciona un rastro al que se desea llegar.

- ¿Cuándo? Nos indica el tiempo de lo sucedido. Entre más detallado es mejor, ya que un evento que sucedió en cierto momento puede ser normal; pero si ocurre dos minutos después, ya podría ser sospechoso. Una característica importante para tener un marco adecuado de referencia del tiempo, es la sincronía; se debe cuidar que las bitácoras se registren en un mismo tiempo. El acceso a un sistema que se registre una hora después de

la modificación de privilegios de un usuario es inconsistente e incluso no se toma en cuenta como evidencia en aspectos legales.

- ¿A través de qué medio? Nos indica el o los medios que se utilizaron para realizar el evento. Podría ser que una tarea programada haya sido la encargada de cambiar la propiedad de un conjunto de archivos de un directorio o que se accedió por un dispositivo de red (*switch*) para habilitar un puerto.

Por lo general, las bitácoras se clasifican de acuerdo a su orientación:

- Aplicación. Indican los sucesos que están activados para registrarse en una aplicación.
- Seguridad. Orientados a dar información sobre seguridad.
- Sistema. Registran actividades del sistema (donde reside la aplicación).

Cualquier aplicación, sistema o parte de *software* o *hardware* específico puede producir una bitácora. Lo importante es que cada elemento de acción genere una.

Cada que un usuario ingresa al sistema, las bitácoras registran las actividades que realizó, por lo que es muy importante proteger los archivos que las contienen para evitar que puedan ser modificados o borrados. La mayoría de los *logs* son almacenados o desplegados en formato ASCII por lo que, los *logs* generados por un dispositivo en particular, pueden ser leídos y desplegados en otro dispositivo diferente.

Propósito de las bitácoras o *logs*

- Son de ayuda en la resolución de problemas.

- Proporcionan avisos tempranos de abuso del sistema.
- Ante una falla o caída del sistema, proporcionan datos de lo ocurrido.
- Pueden ser de utilidad como evidencia legal y en las actividades de auditoría de los sistemas.

Todos los sistemas están expuestos a ser comprometidos de forma local o remota. La seguridad debe enfocarse en la prevención, y en la identificación de amenazas y eventos como los descritos en el tema 6.10; pero también en la identificación de eventos que pueden poner en riego al sistema, por lo que debe realizarse un constante monitoreo a través de las bitácoras de forma completa, cuidadosa, precisa y continua.

Las bitácoras generan mucha información, por lo que es necesario adoptar un sistema automático de monitoreo que envíe las alertas críticas y actuar en consecuencia. Cada sistema operativo cuenta con herramientas que permiten el análisis y monitoreo automático. Por ejemplo, el sistema operativo “Debian” utiliza LogCheck. *Red Hat* utiliza *LogWatch*. En Windows se puede utilizar el programa eventcl.exe para automatizar el visor de sucesos.

Logs remotos

En muchas situaciones es muy importante guardar los registros de máquinas remotas concentrándolos en un solo equipo, de esta forma cuando un equipo sea comprometido y sus *logs* modificados, se pueden seguir registrando las actividades sospechosas en un equipo seguro. Todos los mensajes generados en la máquina origen se enviarán al equipo destino y se registrarán según las políticas del equipo destino: en un directorio, dispositivo o reenviarse inclusive a otro equipo. Esto deberá realizarse correctamente ya que un atacante con un programa “*sniffer*” puede tener acceso a la información de las bitácoras, sobre

todo si se transmiten en texto claro, por lo que es recomendable cifrar las comunicaciones utilizando por ejemplo el protocolo SSH.

Mejores prácticas

Para una buena administración y monitoreo de las bitácoras, se recomienda seguir las siguientes prácticas:

- Protección de bitácoras. Buscar que las bitácoras sean: exactas, autentificables y accesibles.
- Exactos. Registrar todo en las bitácoras, mantener el tiempo real y utilizar múltiples sensores para registrar eventos.
- Autenticidad. Probar que no hayan sido modificados los archivos de las bitácoras, utilizando firmas digitales, encriptación y comprobación *cheksums*.
- Trabajar con copias de bitácoras, almacenar las originales.
- La bitácora creada debe ser auditado y protegido contra accesos no autorizados.
- Respaldar las bitácoras con base en su importancia y tiempo de vida.
- Utilizar herramientas de administración de bitácoras del mismo sistema operativo (*syslog*) o herramientas de terceros; *tripwire*, *logrotate*, etcétera.
- Utilizar herramientas de monitoreo y análisis de *logs*: *LogCheck*, *LogWatch*, CDM, etcétera.

RESUMEN DE LA UNIDAD

Los sistemas operativos son la parte central de los sistemas de cómputo y están expuestos a diversas amenazas que pueden afectar su funcionamiento. Estas amenazas son: la interrupción, intercepción, modificación, fabricación e ingeniería social, por lo que es necesario implementar mecanismos de seguridad informática, la cual se entiende como un conjunto de estrategias y procesos coordinados para la correcta aplicación de mecanismos que permitan mantener la confidencialidad, integridad, autenticidad y disponibilidad de los sistemas de cómputo.

Los mecanismos son de diferente tipo: la criptografía de llave pública y privada, que utiliza diversos estándares, puede aplicarse a necesidades específicas de seguridad, tales como: DES, 3DES, IDEA, RSA, *firmas digitales*, etcétera. Otro de los mecanismos de seguridad es la matriz de acceso que define los derechos de acceso que los sujetos tienen sobre los objetos en el sistema operativo. También existen otros mecanismos de seguridad como los antivirus y las contraseñas de una sola vez.

Ningún sistema está libre de fallas de *hardware*, *software* y errores humanos, por lo que se debe considerar un esquema de reconstrucción ante algún incidente o desastre que ocurra, de manera que se pueda recuperar un equipo a su estado normal de funcionamiento, dando acceso a todos los recursos que administra el sistema.

Las bitácoras son archivos y directorios en los que se registran las actividades realizadas en un sistema durante un tiempo en particular, las cuales hay que saber analizar y respaldar para administrar de manera correcta y segura un equipo de cómputo. Las bitácoras también son importantes en cuestiones de tipo legal.

BIBLIOGRAFÍA

SUGERIDA

Autor	Capítulo	Páginas
Silberschatz, Abraham	19	591-664
Tanenbaum, Andrew	9	584-661
Carretero, Jesús	9	498-538
Microsoft Corporation	12	658-677

UNIDAD 7

IMPLANTACIÓN DE SISTEMAS OPERATIVOS

OBJETIVO PARTICULAR

El alumno identificará las etapas, actividades y características que se deben llevar a cabo para la implantación de un sistema operativo y la configuración para establecer su administración.

TEMARIO DETALLADO

(8 horas)

7. Implantación de sistemas operativos

7.1. El superusuario o administrador del sistema
7.2. Selección del SO (Linux vs Windows NT)
7.3. Preparación de discos de arranque
7.4. Planeación de la utilización de los discos
7.5. Creación del sistema de archivos
7.6. Administración del espacio libre
7.7. Instalación de *Shell*, herramientas y compiladores
7.8. Creación de usuarios y grupos

INTRODUCCIÓN

Después de haber realizado los estudios correspondientes sobre qué sistema operativo deberá instalarse, sigue la etapa de la implantación, que deberá planearse cuidadosamente, considerando aspectos tales como: requerimientos de *hardware* del sistema operativo, compatibilidad con otros componentes, *software* adicional requerido, aplicaciones que correrán en el sistema, bases de datos, infraestructura de red ya instalada o por instalarse, sistema de archivos, uso de licencias, tipos y perfiles de usuario, almacenamiento local o remoto, asignación de espacio en disco, seguridad del sistema operativo, manuales del equipo, discos de instalación, etcétera. En la presente unidad identificarás los criterios para decidir qué sistema aplicar, las etapas y actividades principales para su instalación, así como los aspectos que permitan realizar su administración, que signifique la implementación exitosa de un sistema operativo multiusuario.

7.1. El superusuario o administrador del sistema

La cuenta del superusuario

A la cuenta maestra del administrador de un sistema se le conoce comúnmente como superusuario, *root* o administrador. Estas cuentas vienen preconfiguradas para facilitar la instalación del sistema operativo que se trate. La cuenta de administrador se considera privilegiada, es muy poderosa y peligrosa, ya que se puede hacer prácticamente lo que se quiera con el sistema completo, de tal forma que la mayoría de los ataques a la seguridad de los sistemas se realizan indagando o robando las contraseñas de estas cuentas para hacer cambios, como por ejemplo, modificar los permisos de acceso, borrar archivos, robar información e incluso dañar de forma irreparable el sistema de archivos o el sistema completo. Estas cuentas no deben utilizarse para realizar tareas cotidianas como enviar o recibir *e-mail*, programación o la exploración completa del sistema, para esto se deberán crear cuentas de usuario normales en la que si existe algún error no afecten al sistema. Siempre se debe entender el efecto de aplicar los comandos auxiliándose de la documentación correspondiente, ya que igualmente por errores se pueden causar daños.

Contraseñas

Las cuentas de superusuario o administrador deben tener contraseñas que sean difíciles de descubrir, éstas deben tener al menos 8 caracteres, combinar letras y números o símbolos especiales y combinar letras mayúsculas y minúsculas. Estas contraseñas deberán actualizarse al menos cada tres meses, cada que el administrador cambie, renuncie o lo despidan y siempre que se tengan sospechas de una situación de inseguridad.

Acceso a superusuario o administrador

Se debe deshabilitar el acceso a las cuentas privilegiadas desde todos los equipos, excepto en la consola o terminal del servidor, utilizando cuentas creadas con funciones de *root* para tareas específicas de administración, como son las cuentas **"su"** u otras que defina el administrador. Por ejemplo, "**su** nombre_del_usuario", en este caso, se accede a superusuario por medio de la cuenta del usuario que lo realiza y el acceso queda registrado. El acceso remoto autorizado debe realizarse utilizando programas que cifren las comunicaciones como https, ssh, firmas digitales u otros.

Ética del administrador

La industria de la computación (Flynn, 2001: 268-270) se ha asociado con una falta de comportamiento ético, esto se debe principalmente al retraso de leyes sobre la protección de datos personales electrónico y privado y a los ataques que sufren los sistemas de cómputo. Como se mencionó al inicio de este tema, los administradores de sistemas que utilizan las cuentas de superusuario o administrador tienen más permisos que cualquier otro usuario y que si bien las requieren para realizar su trabajo, llegan a tener acceso a partes críticas, sensibles o a todo el sistema. Es como tener la combinación de una caja fuerte donde se

guarda un gran tesoro. De manera general, se pueden identificar amenazas tales como: el abuso de privilegios, errores, malas configuraciones, *software* ilegalmente instalado, lo que puede derivar en el robo de información, inserción de código malicioso, indisponibilidad del sistema, pago de multas excesivas, etcétera. Actualmente ya existen diversas regulaciones que se deben cumplir o estándares que se deciden seguir y que solicitan de manera concreta una adecuada gestión de usuarios privilegiados como son: Estándar ISO - 27001, Ley General de Protección de Datos Personales en Posesión de Particulares, regulaciones de la CNBV, etcétera. En lo que se refiere al entorno de los fabricantes, ya se encuentran disponibles en el mercado productos que permiten la gestión de usuarios privilegiados. La gestión de usuarios privilegiados permite que los administradores de equipos realicen sus labores bajo un ambiente controlado; es decir, de acuerdo a políticas establecidas, monitorizado y auditado por medio de los siguientes controles.

- Controles administrativos. Son normas que incluyen políticas y procedimientos que definen específicamente quién, qué, cuándo y dónde se pueden utilizar comandos o permisos de superusuario, tratando de apegarse a las buenas prácticas de seguridad como la separación de funciones o de mínimo privilegio, así como el uso de contraseñas robustas.

- Controles tecnológicos y operativos. Es el uso de la tecnología que permite traducir las políticas en mecanismos para definir qué comandos y bajo qué condiciones se ejecutan, para monitorizar las actividades realizadas y poder auditar en cualquier momento las acciones realizadas.

Consideraciones adicionales para la implementación de una estrategia de gestión de usuarios privilegiados y que en algunos casos requieren del uso de la tecnología:

- Realizar un análisis de riesgos del sistema operativo para encontrar puntos que deben ser atendidos.

- Revisar las cuentas de los usuarios privilegiados en los equipos que se adquieren para determinar que no estén configuradas con valores de fábrica o por *default* y que son de fácil acceso.

- En sistemas de servicio crítico combinar los sistemas de gestión de usuarios privilegiados con sistemas de autenticación robusta como son los *tokens* o sistemas OTP (*One Time Password*).

- Considerar el doble control para sistemas sensibles y comandos críticos; es decir, que un acceso o ejecución de comandos no dependa de una sola persona.

- Configurar y proteger las bitácoras para registrar todas las acciones realizadas.

7.2. Selección del SO (Linux VS Windows NT)

Tomar la decisión de qué sistema operativo deberá instalarse en una organización no es una tarea sencilla y se torna más crítica cuando se trata de sistemas operativos multiusuarios para servidores de datos y aplicaciones que trabajan en red como lo son Linux y Windows NT. Son varios los factores a considerar principalmente: la función que realizará el servidor, aplicaciones que soportará, configuración de *hardware*, costo total del sistema y su diseño. Varios de estos

factores se pueden resolver tomando como criterio más importante el diseño y características de cada sistema desarrollados en las unidades 2 y 3 de este material. A continuación se describen de manera concreta:

Sistema Linux

El sistema operativo Linux es un sistema similar a Unix, por lo que contiene las características comunes de Unix, aunque Linux es más joven: su desarrollo comenzó en 1991. Es un **sistema multiusuario, multitarea y de propósito general**.

La estructura de Linux (Carretero, 2001: 613-614.) es de carácter monolítico como la mayoría de los sistemas Unix, a pesar de esto su núcleo es dinámico y abierto ya que se pueden agregar y eliminar módulos de código en tiempo de ejecución. Por ejemplo, se pueden agregar módulos con nuevos tipos de sistema de archivos, nuevos manejadores de dispositivos o gestores de formatos ejecutables. “El sistema Linux completo no sólo está compuesto por el núcleo monolítico; sino que incluye programas del sistema como son los demonios y bibliotecas del sistema. Debido a las dificultades que existen para su instalación y configuración, existen distribuciones que incluyen el núcleo, programas, bibliotecas y herramientas de instalación y configuración que facilitan de manera considerable esta tarea”. Como se mencionó en los capítulos anteriores, existen distribuciones de carácter comercial y gratuito. Las distribuciones más comunes son: Ubuntu, Fedora, CentOS, *Slackware, Debian, openSUSE y Red Hat Enterprise Linux, entre otros*.

¿Por qué seleccionar Linux?

- Sistema operativo moderno, robusto y rápido que puede obtenerse de Internet.

- Existen distribuciones a precios razonables para empresas de cualquier tamaño.
- Contiene un código independiente del procesador, inicialmente se desarrolló para Intel y actualmente puede instalarse en diversas arquitecturas de *hardware* de bajo costo.
- Proporciona una interfaz POSIX (Interfaz portable del sistema operativo), con esto, los diseñadores de *software* pueden ejecutar sus aplicaciones en cualquier sistema operativo compatible con POSIX.
- Basado en los estándares de Unix, por lo que puede correr aplicaciones Unix y cada vez más aplicaciones con soporte comercial.
- Permite incluir de forma dinámica nuevas funciones al núcleo del sistema operativo por medio del mecanismo de módulos.
- Permite ejecutar más de un programa a la vez utilizando uno o varios procesadores.
- Acepta una amplia variedad de dispositivos, que incluye tarjetas de sonido, interfaces gráficos, redes, interfaces SCSI, etcétera.
- Utiliza el *caché* de *buffer* que acepta un área de memoria reservada para amortiguar la E/S de diferentes procesos.
- Permite las particiones de archivo utilizadas por sistemas de archivos como: Ext2 y particiones con otros formatos (MS-DOS, ISO 9060, etcétera.)
- Sistema multiusuario y multiproceso que proporciona protección entre procesos por medio de su planificador de tiempo compartido.
- Puede acceder de forma simultánea a múltiples protocolos de red por medio de la interfaz de *sockets,* lo que permite conectar equipos en red con TCP/IP y otros protocolos.
- Utiliza una capa de abstracción que permite administrar múltiples sistemas de archivos diferentes.
- Soporta sistemas de archivos orientados a dispositivos de red y virtuales.

- Administra la memoria por medio del compartimiento de páginas lo que permite minimizar la duplicación de datos compartidos por diferentes procesos. Esto evita que al ocurrir errores de las aplicaciones detengan el núcleo de Linux.
- Incorpora varios entornos gráficos como: KDE, GNOME, XFCE, etcétera.
- Existen miles de programas libres para Linux de diversos propósitos disponibles en la red para utilizarse con GNU/Linux.
- Existe mucha documentación en la red.
- Tiene un costo total de propiedad muy bajo al instalarse en muchas máquinas.

Sistema Windows NT

Windows NT también se considera un **sistema operativo moderno, multisuario y multitarea de 32 o 64 *bits***. Su diseño inició en 1989 a cargo de la empresa Microsoft y surgió como resultado de unir las ideas ya encontradas en otros sistemas operativos como Unix, VMS, MACH y de la optimización de las mismas. Para mantener la compatibilidad de otros sistemas operativos anteriores a Microsoft, se mantuvieron algunas ideas que existían en MS-DOS y Windows 3.x, por lo que sus principios de diseño se parecen mucho a otros sistemas operativos. Sus características principales son su diseño de tipo micro núcleo, en la que una parte del núcleo da soporte a las funciones de la capa “ejecutivo” del sistema operativo. Dentro del ejecutivo resalta la integración del modelo de seguridad (nivel C2), la gestión de red, la existencia del sistema de archivos grandes, y la aplicación del modelo orientado a objetos.

¿Por qué seleccionar Windows NT?

Por su parte, Carretero expone las características de Windows NT (Carretero, 2001: 620-647).

- Es un sistema multiusuario, multiproceso y de alta difusión comercial. Su utilización requiere el uso de licencias de *software*.
- Cuenta con un avanzado sistema de soporte en *actualizaciones de software.*
- *Se* puede ejecutar en procesadores de tipo: Conjunto de instrucciones complejas (CISC, *complex instruction set computer*) para Intel y *conjunto de instrucciones reducidas* (RISC, *reduced instruction set computer*) para Digital, Alpha, *etcétera.*
- *Es* compatible con POSIX (Interfaz portable del sistema *operativo), OS/2, Win32, MS-DOS y Windows 3.x (Win16).*
- *Contiene* un alto nivel de seguridad C2 de acuerdo al departamento de defensa (DoD) de los EE.UU. La seguridad la aplica a todos los componentes del sistema operativo y a las *aplicaciones externas al mismo.*
- *Es* muy fiable y robusto en el manejo de los procesos y en el sistema de archivos, como: sistemas de archivos con puntos de recuperación, información redundante con técnicas de paridad, técnicas de gestión de memoria y la existencia de *depuradores internos al núcleo.*
- *Proporciona* un procesamiento distribuido por medio de las utilerías de gestión de redes como parte del núcleo del sistema; protocolos de transporte, *sockets*, colas de mensajes*, etcétera.*
- Es muy eficiente tanto en monoprocesadores como en multiprocesadores, a través de su modelo de procesos ligeros y un sistema de entrada/salida muy compacto, en el que todos sus componentes se utilizan como manejadores de dispositivos.

7.3. Preparación de discos de arranque

Antes de realizar la instalación de cualquier sistema operativo, se debe determinar si se actualizará la instalación ya existente o bien si se llevará a cabo una nueva. Una actualización instala de manera automática una versión del sistema operativo en el mismo directorio del sistema originalmente instalado. Una instalación, en contraste con la actualización, es el proceso de ubicar el sistema operativo en un nuevo directorio del disco duro o partición definida por el administrador, en el que no exista algún sistema previamente instalado. Si ya existiera, éste se borrará. La preparación y utilización de los discos de instalación o actualización dependerá del tipo de sistema operativo (*software* libre o comercial), los métodos de instalación cambian rápidamente por el desarrollo tecnológico. A continuación se describen los más comunes.

Sistema Linux

La forma más sencilla de preparar los discos de instalación de Linux es la siguiente:

1. Ingresar al sitio http://www.linux.org/dist/list.html y elegir la distribución correspondiente.
2. Bajar la imagen ISO del sistema operativo y grabarla en un CD/DVD.
3. Etiquetar los discos (Distribución, fecha, orden consecutivo).
4. Configurar el BIOS del servidor para que arranque desde el CD/DVD.
5. Arrancar equipo, elegir instalación gráfica y seguir el asistente.
6. Proporcionar la información que solicita durante la instalación (previamente preparada), particiones, sistema de archivos, etc.
7. Continuar con la instalación.

Sistema Windows NT

Como se comentó anteriormente, Windows NT es un sistema operativo comercial, para su uso se requiere de la compra de licencias que tienen un costo, dependiente de la versión del sistema operativo, modo de licencias y número de usuarios que se conectarán al equipo. Existen dos tipos básicos de licencias:

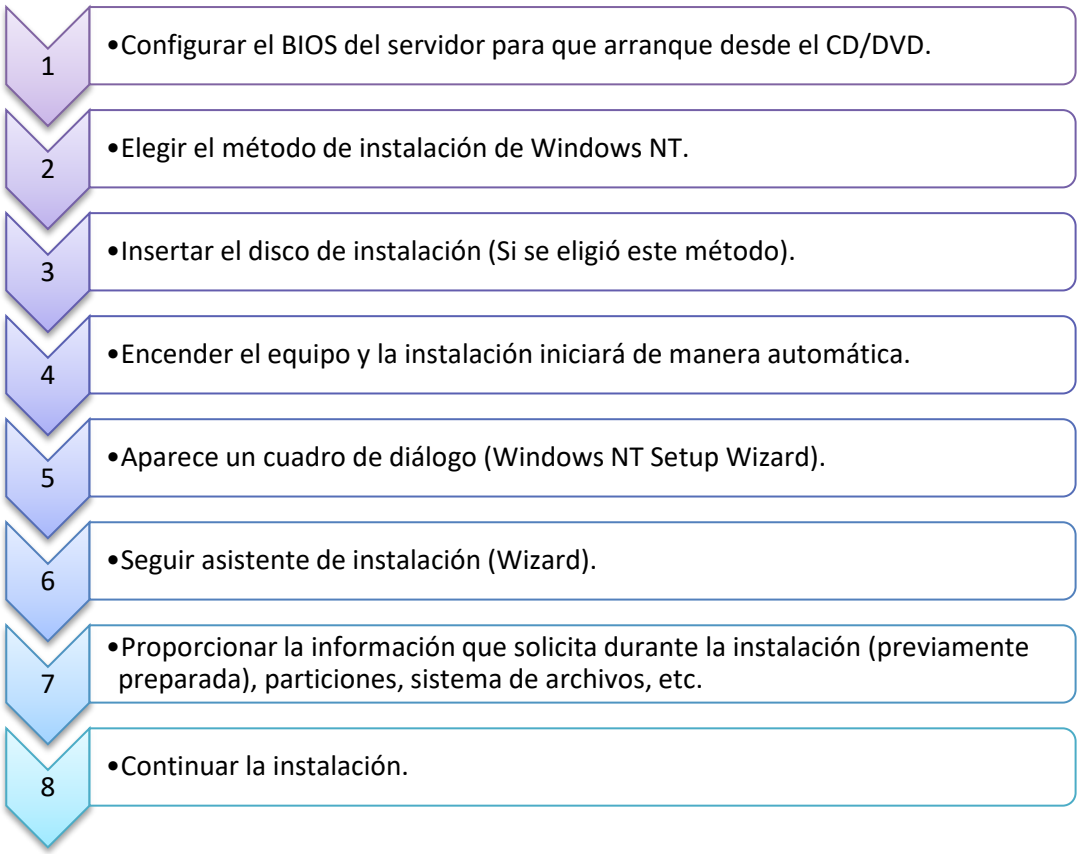
1. Por servidor. Cada conexión simultánea al servidor debe tener una licencia de cliente separada.
2. Por sitio. Cada computadora que acceda a cualquier servidor en la red debe tener su propia licencia individual de cliente.

Sin importar la opción de instalación que se elija para instalar Windows NT Server, se debe ejecutar Winnt.exe o Winnt32.exe. Se puede utilizar el programa Setup.exe para iniciarlos, el programa de instalación Setup.exe, está ubicado en el directorio raíz del CD/DVD de instalación. En equipos que ya tienen instalado alguna versión de Windows, se puede ejecutar directamente desde la línea de comando MS-DOS los programas Winnt.exe o Winnt32 sólo hay que verificar los manuales de la versión que se va a instalar o actualizar. Los discos de instalación se entregan al adquirir el sistema operativo y se instala de forma similar a Linux.

Opciones de instalación

- Instalación nueva. Disquetes de instalación y unidad de CD-ROM local. Este tipo de instalación fue muy común, actualmente ya no se utiliza.
- Instalación nueva. Sin disquetes de instalación y con unidad de CD/DVD local. Si el equipo cuenta con una unidad de CD/DVD local se inicia la instalación, sin tener que pasar por el proceso de crear los discos de instalación. Este tipo de instalación es muy rápida y ahorra un tiempo considerable. Si se tiene instalado un sistema Microsoft Windows NT 3.51 se puede ejecutar la siguiente instrucción, en donde la unidad de CD/DVD está definida con la letra E y el procesador es Alpha: E:\ALPHA\WINNT32.EXE /B
- Instalación nueva. Unidad de disquetes y unidad de CD/DVD de red. Si el servidor no cuenta con una unidad de CD/DVD local, pero tiene una red instalada, se puede instalar Windows NT Server utilizando un DVD-ROM que se comparta a través de la red. Se asigna a un DVD-ROM de la red una letra de unidad y se instala el sistema. Por ejemplo si el servidor cuenta con procesador Intel y se asigna la letra D a la unidad, se ejecuta: D: \I386\WINNT.EXE /B. El programa investigará si existe suficiente espacio libre en el disco.
- Instalación de actualización. Las opciones son similares a las anteriores con excepción de que Windows NT tomará las decisiones para la instalación. (Microsoft Corporation, 2001: 53-59).

Instalación.



Nota: Durante el proceso de instalación, es muy importante tener a la mano los manuales de instalación correspondientes para resolver cualquier duda o problema que se llegare a presentar.

7.4. Planeación de la utilización de los discos

Una de las actividades más críticas durante la instalación de un sistema operativo es determinar el espacio que tendrán los discos duros, esto requiere de una planeación previa que considere principalmente la función que tendrá el servidor, tipo y versión o distribución del sistema operativo, sistema de archivos, crecimiento, etcétera. La asignación del espacio de los discos se lleva a cabo por medio del proceso denominado “partición”, que consiste en dividir un disco físico en varios discos lógicos; es decir, se divide el disco en varias secciones que son independientes entre sí. Existen dos tipos de particiones; primarias y extendidas. La partición primaria es en la que se instala el sistema operativo para que éste pueda ejecutar “*boot*”. La partición extendida es un área de almacenamiento. En un disco sólo se podrán tener cuatro particiones primarias o tres primarias y una extendida. En la partición extendida se podrán definir todas las unidades lógicas que se requieran. Cualquier computadora que cuente con un disco duro tiene al menos una partición primaria con una capacidad de almacenamiento igual al tamaño del disco.

Ventajas que proporciona la partición:

- Es excelente cuando se planea instalar más de un sistema operativo en el mismo disco.
- Si existe un error en alguna partición, las otras no se afectarán.
- Se pueden almacenar los datos en particiones independientes.
- Se puede borrar o cambiar el contenido de una partición sin que afecte a las otras.

Las particiones se pueden realizar de manera manual o automática y deben adecuarse al tipo de sistema operativo que se decidió instalar. Para que la instalación sea **exitosa y segura**, es necesario contar con toda la información necesaria y elaborar un plan específico con anticipación y cuidado de acuerdo a los requerimientos del sistema operativo y la función que realizará el servidor. El plan deberá contener todos los elementos que soportará el o los discos duros, lo que permitirá eliminar problemas antes y después de realizarla, tales como interrumpir la instalación en curso, terminarla erróneamente e inclusive iniciar de nuevo todo el proceso, lo que no es recomendable; además de que consumirá demasiado tiempo. A continuación se muestran los principales aspectos a considerar por medio de una lista de comprobación que puede variar de acuerdo al sistema a instalar.

Lista de comprobación:

Pregunta	Respuesta
¿Qué función realizará el servidor?	
¿Qué sistema operativo se instalará? (versión, distribución, etcétera.)	
¿Se realizará una actualización del sistema operativo?	
¿El equipo cuenta con los requerimientos mínimos de *hardware* para instalar el sistema y sus aplicaciones?	
¿Qué tipo de controlador de unidad de disco se utilizará?	
¿El sistema es compatible con todo el *hardware* del equipo? (interfaces de red, tarjetas de sonido, controladores de CD-ROM, etcétera)	
¿Se requiere un controlador de arranque especial del fabricante?	
¿Qué partición será la partición principal de arranque?	
¿Qué partición será la partición del sistema?	
¿Qué sistema de archivos se instalará?	
¿Qué tamaño se asignará al *swap*?	
¿Qué modo de licencia se instalará? En el caso de Windows NT.	
¿Cuáles aplicaciones se instalarán?	
¿Qué nombre que se dará al servidor?	
¿Qué *password* se asignará al equipo?	

¿Cuál es la marca/modelo de las interfaces de red?	
¿Cuál es el tipo de red a la que se conectará el equipo? (*Ethernet, Gigabit Ethernet*, etcétera)	
¿Cuáles protocolos de red se instalarán? (TCP/IP, NetBEUI, IPX/SPX, otros)	
¿Cuáles servicios de red se instalarán?	
¿Se instalará una dirección IP o un servidor DHCP?	
¿Qué parámetros de configuración de red se utilizarán? (IP, máscara, ruteo, dns, dominio, otros)	
¿Cuál es el método de instalación del sistema? (discos, CD-ROM, red)	

Sistema Linux

Después que se ha determinado la función que tendrá el servidor y los principales aspectos descritos en la lista de comprobación, se podrá definir el tipo, tamaño y número de particiones que tendrán el o los discos duros. En el sistema Linux (a diferencia de Windows NT en donde se accede a los dispositivos de almacenamiento a través de símbolos como a: c: d: etcétera) cada disco o dispositivo, al ser montado, se asocia a un directorio del disco duro para que el sistema operativo lo reconozca. Para una planeación adecuada, Linux requiere al menos dos particiones: una para el sistema y los datos y otra para el "*swap*" (extensión de memoria virtual); aunque es común que se utilicen tres: una para el sistema y programas (*/*), otra para los datos (**/home**) y otra para **swap**.

Particiones recomendadas:

Las particiones pueden crearse por método manual utilizando el comando “fdisk” o bien de manera automática “*Disk Druid*” en el momento de la instalación, por lo que debe planear el tamaño de *swap* con anticipación.

Las particiones pueden ser:

- Para sistemas pequeños y pocos usuarios, dos o tres particiones, esto permitirá tener espacio disponible para particiones futuras.

- Para sistemas grandes y gran cantidad de usuarios, varias particiones y discos duros.

Swap

Otro de los elementos que se debe considerar es el tamaño de la partición para “*swap*”. La *swap* es un espacio reservado en el disco duro que se utiliza como una extensión de la memoria RAM de un sistema y que permite creer a los programas que existe más memoria de la que realmente se tiene. El sistema operativo se encarga de pasar los datos a la *swap* cuando se requiere más espacio libre en la memoria RAM y viceversa (proceso de intercambio). En Linux la memoria total del sistema está formada por la memoria RAM instalada más la *swap* disponible. Es importante mencionar que el acceso a la *swap* (disco duro) es más lento que el acceso a RAM; por lo que, si el equipo está muy cargado de trabajo y se hace un uso intensivo de *swap*, la velocidad del sistema disminuirá. No existe una fórmula para determinar cuánto espacio se requiere disponer para *swap*; sin embargo, existen algunas recomendaciones generales:

- El área de *swap* debe ser dos veces el tamaño de la memoria RAM.
- En sistemas con mucha memoria, instalar espacio para *swap*.
- Si se cuenta con más de un disco, instalar *swap* en el disco más rápido.
- Si se cuenta con más de un disco con acceso simultáneo, instalar una partición *swap* en cada uno de ellos.

Igualmente, Linux requiere que exista el directorio raíz / para la distribución de sus directorios más importantes y que el administrador asigne el espacio requerido del espacio en el disco.

Directorios

- / Directorio raíz (5 GB).
- /boot Almacena programas de arranque (100 MB).
- /usr Programas del sistema y programas fuente (1000 MB).
- /home Directorios de los usuarios (depende de cada uso).
- /opt *Software* de terceros (depende de la cantidad de *software*).
- /tmp Archivos temporales.
- /usr/local *Software* GNU adicional (500 MB).

Sistema Windows NT

Como ya se mencionó, Windows NT identifica las particiones con letras del abecedario (C, D, E, etcétera); su tamaño depende de la versión del sistema operativo, así como de la marca, modelo y configuración del adaptador del disco duro. Por ejemplo, la versión de Windows NT 2000 Server necesita una partición de inicio de al menos 671 MB de espacio libre para instalar todos los archivos del sistema operativo; sin embargo, siempre se recomienda crear particiones de inicio de al menos 2 GB para permitir que en el futuro se instalen archivos y programas adicionales.

Configuración de discos.

Antes de instalar Windows NT Server en un disco duro, la parte del disco que Windows NT utilizará, se debe iniciar con un tipo de almacenamiento, con partición que se deberá formatear. Si el sistema y la partición de inicio se separan, se tiene que asignar particiones y formatear tanto al área del disco que contendrá los archivos del sistema como al área del disco que contendrá el sistema operativo. Las tareas que deben realizarse para preparar un disco son:

- Inicialización del disco. La inicialización consiste en asignar un tipo de almacenamiento que define la estructura fundamental de un disco duro. Existen dos tipos de almacenamiento de disco: básico y dinámico.

- Creación de particiones sobre un disco básico o volúmenes sobre un disco dinámico.

- Dar formato al disco. Después de crear una partición, se debe formatear con un sistema de archivos específico NTFS o alguno de los sistemas de archivos FAT16 o FAT32 (Microsoft, 2001, p. 126-131).

Terminología

Es importante entender los tipos de almacenamiento, tipo de particiones y tipos de volúmenes disponibles. Se explican a continuación:

- **Almacenamiento básico.** Es la forma estándar de almacenamiento y consiste en dividir un disco duro en particiones. Recordemos que la partición es una parte del disco que funciona como una unidad de almacenamiento físicamente separada. Windows NT Server reconoce particiones primarias y extendidas. Un disco que se inicia para almacenamiento básico se denomina también disco

básico y puede contener particiones primarias, particiones extendidas y unidades lógicas. Cuando se añade un disco nuevo a un equipo con Windows NT Server se le considera a éste un disco básico. La siguiente figura muestra el almacenamiento básico que contiene cuatro particiones primarias que a su vez contienen particiones extendidas.

- **Almacenamiento dinámico**. Windows NT Server soporta este tipo de almacenamiento y consiste en crear una sola partición que incluye el disco entero. Un disco que se inicializa para almacenamiento dinámico es un disco dinámico. Los discos dinámicos se dividen en volúmenes. Un volumen consiste en una parte o partes de uno o más discos físicos. Un disco dinámico puede contener los siguientes tipos de volúmenes:

  - ✓ Volúmenes simples. Contiene espacio de un solo disco y no es tolerante a fallas.

  - ✓ Volumen distribuido. Incluye espacio de disco de varios discos. La cantidad de discos soportados depende del tipo de sistema operativo Windows NT. Por ejemplo, Windows 2000 Server soporta hasta 32 discos.

  - ✓ Volumen con espejo. Consiste en dos copias idénticas de un volumen simple, cada una en un disco por separado. Este tipo de volumen proporciona tolerancia a fallas en caso de que falle un disco duro.

  - ✓ Volumen seccionado. Combina áreas de varios discos duros para proporcionar un tamaño de volumen mayor para un mejor rendimiento y es tolerante a fallas. Tiene diferentes niveles (Raid - 0, Raid - 1, Raid – 5, etcétera).

Particiones

Recordemos que un disco básico se puede dividir en particiones primarias y extendidas y la partición nos permite separar diferente tipo de información como los datos del usuario en una partición y las aplicaciones en otra. Un disco básico puede contener hasta cuatro particiones primarias, o hasta tres particiones primarias y una extendida y solamente una partición puede ser una partición extendida.

- Particiones primarias. Windows NT Server utiliza las particiones primarias para iniciar un servidor y una de estas particiones se marca como activa. La partición activa contiene los archivos de inicio del sistema operativo y sólo una partición en un disco puede estar activa a la vez. Varias particiones primarias permiten separar diferentes sistemas operativos o algún tipo de datos.

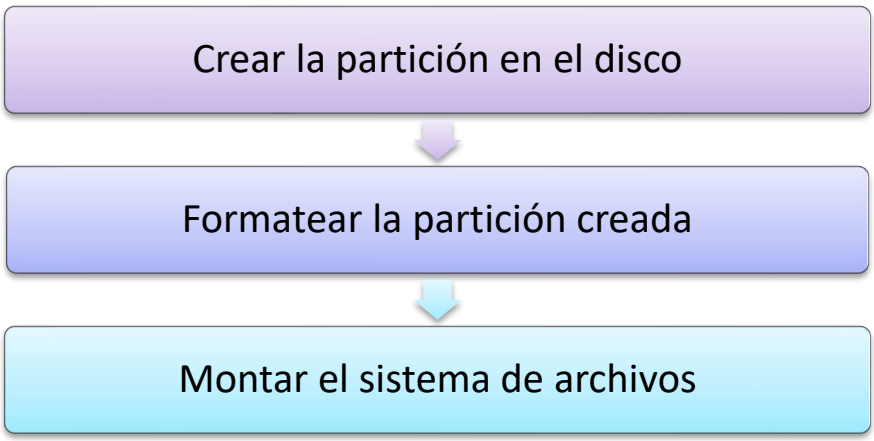
- Particiones extendidas. Se crea una partición extendida a partir del espacio libre del disco. Sólo puede existir una partición de este tipo en un disco duro, por lo que es importante asignar todo el espacio libre restante en la partición extendida. A este tipo de particiones, no se les da formato, ni se asignan letras de unidad; además, se dividen en segmentos. Cada segmento es una unidad lógica. Se asigna una letra de unidad a cada unidad lógica y se formatea con un sistema de archivos: NTFS, FAT16 y FAT32.

Existen herramientas como "*Disk Management*" para la administración de discos que proporciona una ubicación central para la información de discos y tareas de gestión para crear y borrar particiones y volúmenes y con los permisos adecuados se pueden gestionar discos locales y remotos.

Posterior a este proceso, se asigna el sistema de archivos que se explica en el siguiente tema.

7.5. Creación del sistema de archivos

Recordemos que el sistema de archivos proporciona el mecanismo para el almacenamiento y el acceso en línea de los datos y programas de un sistema operativo y usuarios del mismo, que posteriormente serán representados de forma textual o gráfica utilizando el gestor de archivos. Es importante mencionar que durante la instalación de un sistema operativo, se crea de manera automática o por elección del administrador del equipo, un sistema de archivos compatible al sistema que se está instalando. Por ejemplo, Linux o Windows NT. Sin embargo, cuando éste no existe, es posible instalarlo de manera manual utilizando los comandos o herramientas correspondientes. La elección del sistema de archivos depende del sistema operativo que se vaya a instalar, para su creación se deben realizar las siguientes acciones:



El tema de la partición se describió en el punto anterior. En cuanto al proceso de formatear un disco, es importante aclarar que existen dos tipos de formatos; de bajo y alto nivel. El formato de bajo nivel, también llamado físico, consiste en proporcionar las características físicas al disco, tales como colocar marcas en la superficie (óxido metálico magnetizable) para dividirlo en pistas y sectores con un

determinado tamaño, el tamaño estándar de un sector es de 512 *bytes*, este tipo de formato se realiza con *software* y lo realiza el fabricante. El formato de alto nivel, también llamado lógico es realizado por los administradores o personas que desean instalar un determinado sistema de archivos, en algunos casos este tipo de formato también lo realiza el fabricante. Si ya existiera un sistema de archivos, el nuevo formateo eliminará los datos que existan en el disco completo o partición correspondiente. En el siguiente sitio, se muestra una tabla con los sistemas de archivos más comunes: http://es.ccm.net/contents/611-el-sistema-de-archivos

Sistema Linux

Creación del sistema de archivos

En GNU/Linux el sistema de archivos se puede crear por medio de herramientas gráficas; para el escritorio KDE existe la utilería “QtParted” y para los escritorios GNOME la herramienta “Gparted”, o bien de forma manual a través del comando **“mkfs”** con sus parámetros correspondientes, revisar el manual en línea (#man mkfs) de acuerdo al sistema que se quiera crear. Por ejemplo, si se tiene que instalar el sistema de archivos ext3 en la segunda partición del primer disco, la instrucción será:

#mkfs –t ext3 /dev/hda2

Tipos de dispositivos y particiones en Linux:

Comando	Acción
Disquette No. 1	/dev/fd0
Disquette No. 2	/dev/fd1
Primer disco duro, todo el disco	/dev/hda
Primer disco duro, partición primaria 1	/dev/hda1
Primer disco duro, partición primaria 2	/dev/hda2
Primer disco duro, partición primaria 3	/dev/hda3
Primer disco duro, partición primaria 4	/dev/hda4
Primer disco duro, partición lógica o extendida 1	/dev/hda5
Primer disco duro, partición lógica o extendida 2	/dev/hda6
Segundo disco duro, todo el disco	/dev/hdb
Segundo disco duro, partición primaria 1	/dev/hdb1
Primer disco duro SCSI, todo el disco	/dev/sda
Primer disco duro SCSI, partición primaria 1	/dev/sda1
Segundo disco duro SCSI, todo el disco	/dev/sdb
Segundo disco duro SCSI, partición primaria 1	/dev/sdb1

Montar el sistema de archivos

Posterior a la creación del sistema de archivos, es necesario llevar a cabo el proceso de montaje para que éste pueda ser utilizado. Recordemos que el sistema Linux utiliza un sistema jerárquico de directorios, por lo que el montaje proporciona el mecanismo que integra la jerarquía de directorios para que sea accesible. El montaje se realiza con el comando **"*mount*"**, también se recomienda consultar el

manual en línea (#man mount). Es necesario conocer qué partición del disco o dispositivo (CD-ROM, floppy, etcétera) y qué tipo se desea montar.

Su sintaxis es: **#mount −t [tipo] /dev/[dispositivo] /punto/de/montaje/**

Por ejemplo, se requiere montar el dispositivo /dev/hda1, que corresponde a la partición donde se encuentra la instalación de Windows, en un punto de montaje previamente creado, /mnt/windows. El comando es: **#mount −t vfat /dev/hda1 /mnt/Windows,** puede observarse que el comando indica: tipo de sistema de archivos, dispositivo físico y el directorio desde donde será accesible (punto de montaje).

Para acceder a los directorios de la partición se utiliza: **cd /mnt/windows/** y, posteriormente, el comando **ls –l** para ver el contenido. Para desmontar la unidad se utiliza el comando **“umount”**, para que pueda realizarse es necesario que la unidad esté desocupada y que ningún programa o proceso se encuentren activos, para salir se utiliza el comando **“cd ~”** y posteriormente **“umount /mnt/windows/”.**

Los tipos y características de los sistemas de archivos GNU/Linux que se pueden montar están descritos el tema 3.3 Sistemas de archivos de este material.

Sistema Windows NT

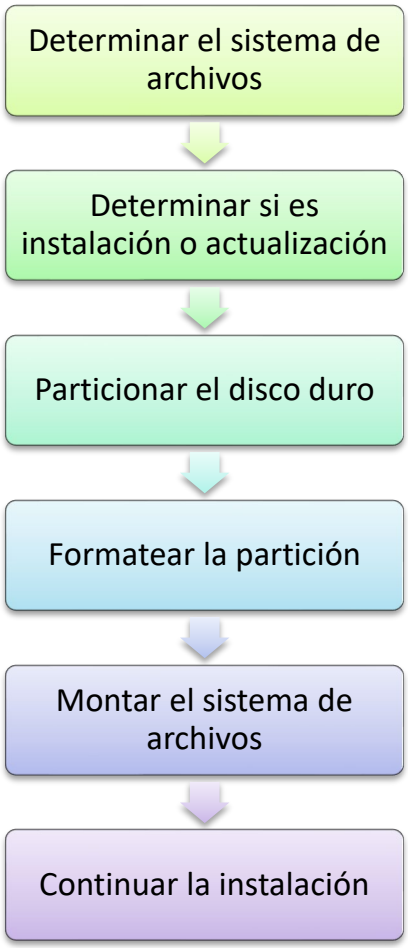
En la unidad 2 de este material se realiza una explicación más amplia del sistema de archivos NTFS; sin embargo, es importante recordar que Microsoft recomienda instalar el sistema de archivos NTFS, excepto cuando se requieran configuraciones de inicio con sistemas operativos diferentes a Windows NT Server. NTFS proporciona accesos más rápidos que FAT, minimiza el número de accesos al disco requeridos para la búsqueda de archivos, proporciona mayor seguridad de archivos,

directorios, grupos y usuarios, así como la opción de utilizar *Active Directory,* entre otros. Al igual que en otros sistemas operativos, antes de la instalación se debe particionar y dar formato con el sistema de archivos NTFS o uno de los sistemas FAT16 o FAT32. El sistema de archivos que se elija afectará las operaciones del disco duro; tales como control de acceso del usuario a los datos, almacenamiento de datos, capacidad del disco y sistemas operativos que accedan a los datos del disco.

Consulta el siguiente sitio, en él se puede ver un ejemplo de comparación de los sistemas de archivos:

http://publib.boulder.ibm.com/html/as400/v4r5/ic2931/info/RZAHQFILESYSTEMS.HTM

La implementación del sistema de archivos se realiza durante la instalación del sistema operativo con los discos de instalación correspondientes y con el plan de instalación previa (lista de comprobación). Los pasos son los siguientes:



El proceso de montaje es sencillo y directo, durante la instalación, se le proporciona al sistema operativo el nombre del dispositivo y la ubicación dentro de la estructura de archivos en la que se ubicará el sistema de archivos (punto de montaje) puede utilizarse la herramienta "*Disk Management*" descrita anteriormente, de igual forma, si ya existe el sistema de archivos FAT, se puede convertir a NTFS ejecutando desde la línea de comandos MS-DOS el comando: convert E/fs:ntfs.

7.6. Administración del espacio libre

En esta unidad se explican dos aspectos: en primer lugar las funciones que realiza el sistema operativo para gestionar el espacio libre en los discos duros, y posteriormente, las herramientas que el administrador del equipo puede utilizar para complementar la gestión de dicho espacio en los sistemas operativos multiusuario. El sistema operativo mantiene una **lista de espacio libre** que registra todos los bloques del disco que están libres; es decir, que no están asignados a algún archivo o directorio. Cuando el sistema crea un archivo, busca en la lista la cantidad de espacio requerido y entonces le asigna el que requiere ese archivo. Cuando se borra un archivo, el espacio liberado se agrega a la lista para reutilizarlo. Casi todos los sistemas de archivos dividen los archivos en bloques de tamaño fijo e internamente implementan los siguientes mecanismos para la gestión de espacio libre (Silberschatz, 2002: 386-388):

- Mapa de *bits*. La lista de espacio libre se implementa como un mapa de *bits* en la que cada bloque se representa mediante1 *bit*. Si el bloque está libre, el *bit* es 1. Si el bloque está asignado, entonces el bit es 0. Muchas computadoras incluyen instrucciones para la manipulación de *bits* que pueden utilizarse para este fin. Por ejemplo, la familia Intel a partir del procesador 80386 y la familia Motorola a partir del procesador 68020, tienen instrucciones que devuelven el desplazamiento en una palabra del primer *bit* con el valor 1.

- Lista enlazada. Este mecanismo consiste en enlazar todos los recursos (bloques o descriptores de archivos) manteniendo un apuntador al primer elemento de la lista. Este apuntador se mantiene siempre en memoria. Cada

elemento de la lista apunta al siguiente recurso libre de este tipo. Cuando el servidor de archivos necesita recursos libres, recorre la lista y desenlaza elementos que ya no están libres. Este método no es muy eficiente, excepto para discos muy llenos y fragmentados, donde las listas de bloques libres son muy pequeños. Este enfoque requiere de mucho proceso de entrada/salida. Un ejemplo de este mecanismo es el sistema de archivos FAT del sistema operativo MS-DOS.

- Agrupación. Es una modificación de la lista enlazada y consiste en almacenar las direcciones de *n* bloques libres en el primer bloque libre. Los primeros *n – 1* de estos bloques están efectivamente libres. El bloque final contiene las direcciones de otros en bloques libres y así sucesivamente. Una de las ventajas de este mecanismo, es que se pueden encontrar rápidamente las direcciones de un gran número de bloques libres a diferencia del enfoque de lista enlazada. Los sistemas Unix y Windows NT utilizan este método.

- Conteo. Este mecanismo consiste en aprovechar que varios bloques contiguos o cercanos unos de otros puedan ser asignados o liberados de forma simultánea, de manera particular cuando se asigna espacio con el algoritmo de asignación contigua o mediante agrupamientos. Por lo que, en lugar de mantener una lista de direcciones de disco libres, se puede mantener la dirección del primer bloque libre y el número de *n* bloques contiguos libres que siguen al primer bloque. Por lo que cada entrada en la lista de espacio libre consiste entonces en una dirección de disco y una cuenta, la lista global será más corta, siempre y cuando la cuenta sea generalmente mayor a 1.

- Cuotas de disco. Para evitar que los usuarios ocupen demasiado espacio de disco, los sistemas multiusuario tienen un mecanismo para asignar cuotas de disco. En este caso el administrador del sistema es quien asigna a cada

usuario una cantidad máxima de archivos y bloques y el sistema operativo cuida que los usuarios no excedan su cuota.

Herramientas

Como se mencionó al inicio, los sistemas operativos multiusuario como Linux, Unix o Windows NT, cuentan con diversas herramientas que el administrador de un equipo puede utilizar para diagnosticar problemas de disco, mejorar su rendimiento y administrar el espacio disponible. En cualquier distribución de Linux se pueden utilizar los comandos **du** y **df**. El comando **du** permite conocer el espacio ocupado en el disco de un archivo o directorio. Su sintaxis es: **du [opciones] [directorio]**.

Las opciones más comunes son:

- -a, --all Proporciona el espacio ocupado de cada archivo.
- -b, --bytes Muestra la información de salida en *bytes*.
- -c, --total Muestra la ocupación total.
- -s Informa la suma total de cada uno de los archivos especificados.
- -k Proporciona la información en Kilobytes (1024) octetos.
- -m Muestra la cantidad en bloques de megabytes.
- -x Excluye los directorios que estén en sistemas de archivos diferentes.

Ejemplo: **du –a images** (muestra el tamaño de cada archivo del directorio images).

El comando **df** muestra el espacio disponible en el disco para cada partición instalada. Su sintaxis es: **df [opciones]**.

Las opciones más comunes son:

- -a Incluye sistemas de archivos falsos.
- -h 200 Muestra los tamaños en formato legible para las personas (1K, M, 3G, etcétera).
- -i Lista información de nodos-i en vez de bloques.
- -l Limita el listado a sistemas de archivos locales.
- -P Utiliza el formato de salida POSIX.
- -T Muestra el tipo de sistema de archivos.

Ejemplo: usuario@maquina:~/$ **df**

Filesystem	1k-blocks	Used	Available	Use%	Mount
/dev/hda2	2949060	2102856	696400	75%	/
/dev/hda1	23302	2593	19506	12%	/boot
/dev/hda4	10144728	5506796	4637932	54%	/home
/dev/hdb2	3678764	3175268	503496	86%	/u

Los comandos descritos se pueden utilizar directamente para administrar el sistema de archivos de Linux. También existen herramientas gráficas que ayudan al administrador a realizar estas tareas de manera más fácil como: *Paragon NTFS for Linux, QT Parted, TestDisk, Gparted, Partimage.*

El sistema operativo Windows NT tiene varias herramientas que permiten diagnosticar problemas de disco (***chkdsk***), mejorar su rendimiento (*Disk Defragmenter*), compresión de datos y cuotas de disco.

- Comprobar disco. La herramienta ***chkdsk*** permite comprobar si existen errores en el sistema de archivos y sectores defectuosos y repararlos. Se

puede realizar desde el explorador (Explorer) de Windows o desde Mi PC (*My Computer*).

- Defragmentador de disco. Windows NT Server almacena archivos y directorios en el primer espacio disponible del disco y no necesariamente en un área de espacio contiguo, por lo que los archivos y directorios se fragmentan. Cuando el disco contiene muchos archivos y directorios fragmentados, el equipo tarda más en tener acceso a ellos ya que requiere varias lecturas adicionales para recoger las distintas partes. La creación de nuevos archivos y directorios también requiere más tiempo, porque el espacio libre disponible está disperso. La herramienta *Disk Defragmenter* se utiliza para ubicar archivos y directorios defragmentados, esta acción la realiza mediante el desplazamiento de las partes de cada archivo o directorio a una ubicación, de modo que cada archivo o directorio ocupa un área de espacio contiguo en el disco. Esta herramienta también consolida espacio libre haciendo menos probable que los archivos nuevos sean fragmentados. La defragmentación la realiza sobre volúmenes FAT16, FAT32 y NTFS. Se accede a esta herramienta por medio del complemento Administrador de equipos (*Computer Management*) o creando una consola personalizada que contenga este complemento.

- Compresión de datos. La compresión de datos permite que los archivos y directorios ocupen menos espacio en volúmenes con formato NTFS. Lo que permite almacenar más datos. La compresión se puede establecer por medio del explorador (Explorer) de Windows o mediante el uso de la utilería **compac** en la línea de comandos. Si los archivos y directorios están cifrados, no puede realizarse la compresión.

- Cuotas de disco. Las cuotas de disco permiten gestionar el crecimiento de almacenamiento en ambientes distribuidos. Las cuotas permiten asignar

espacio de disco a los usuarios basándose en sus propios archivos y directorios. La asignación de cuotas, avisos y límites de cuota se puede habilitar para todos los usuarios o usuarios individuales por medio del cuadro de diálogo Propiedades (*Properties*) de un disco en particular, al pulsar la pestaña Cuota (*Quota*) y configurar las opciones correspondientes.

7.7. Instalación de *Shell*s, herramientas y compiladores

Los entornos gráficos (GUI) para trabajar con los sistemas operativos han evolucionado mucho en los últimos años y prácticamente se puede administrar todo el sistema por medio de estos entornos. Sin embargo, si se desea un mejor dominio, conocer a fondo el sistema y tener una respuesta más rápida, se recomienda aprender a trabajar desde el intérprete de comandos conocido como *Shell*. Este intérprete es una interfaz entre el usuario y el sistema operativo; es decir, se utilizan comandos del sistema operativo en un intérprete de comandos y el *Shell* los interpreta y ejecuta. Los administradores experimentados pueden escribir *scripts* (archivos de comandos) de *Shell* para aumentar sus capacidades. Cuando el *Shell* se pone en marcha, se inicializa y luego muestra en la pantalla un indicador de comandos (*prompt*), comúnmente es un signo de dólar o porcentaje.

Linux

De acuerdo con los planteamientos de Tanenbaum (2003: 683-688), en Unix/Linux, a través del tiempo, se han escrito muchos *Shell*s (sh, ksh, bash, etcétera). Cuando

un usuario teclea un comando, el *Shell* extrae la primera palabra de la línea, lo busca y si lo encuentra lo ejecuta (programa, comando, etcétera) y en ese momento el *Shell* se suspende a sí mismo hasta que se termina y trata de leer la segunda línea de comandos. Lo importante es entender que el *Shell* es un programa de usuario común y lo único que se requiere es la capacidad de leer de la terminal, escribir en ella y que tenga, en ciertos casos, los privilegios de ejecutar otros programas y comandos, ya que si desea programar un *Shell* y no tiene ciertos permisos, el *Shell* no funcionará de acuerdo a sus necesidades. Los comandos pueden tener argumentos que se pasan al programa invocado en forma de cadena de caracteres. Por ejemplo, la línea de comando **cp archivo1 archivo2** invoca el programa **cp** con dos argumentos: archivo1 y archivo2. El programa interpreta como un archivo existente **archivo1**, y crea una copia de ese archivo y le asigna el nombre **archivo2**. No todos los argumentos son nombres de archivo. En **head -30 archivo**, el primer argumento, -30, le indica a *head* que imprima las primeras 30 líneas de archivo, en lugar del número de líneas predeterminada que es 10. Los argumentos que controlan el funcionamiento de un comando o definen un valor opcional se llaman **indicadores**, y por convención se indican con un guion para evitar ambigüedades. Por ejemplo, el siguiente comando ***head* 20 prueba** es válido e indica a *head* que imprima las primeras 10 líneas del archivo nombrado 20 y luego imprima las primeras 10 líneas del archivo prueba. Casi todos los comandos aceptan múltiples indicadores y argumentos. Para facilitar la especificación de varios nombres de archivo, *Shell* acepta caracteres “comodines”. Por ejemplo, el uso del carácter asterisco **(*),** que se entiende como *todo*, el mandato **ls *.c** solicita a **ls** que muestre una lista de todos los archivos cuyo nombre termine con **.c**.

Existen otros caracteres comodines como el signo de interrogación que coincide con cualquier carácter individual, el uso de caracteres entre corchetes que indica que puede seleccionarse cualquiera de ellos. Cuando se inicia el *Shell* se generan de manera automática a un archivo llamado **entrada estándar** (para leer), otro llamado **salida estándar** (para escribir las salidas normales) y otro llamado **error estándar** (para escribir mensajes de error), las lecturas estándar se toman del

teclado y las escrituras de salida estándar o error estándar se envía a la pantalla. También existe el caso de que un programa que toma sus entradas de la entrada estándar y realiza un proceso con ellas y escribe sus salidas en la salida estándar denominada **filtro** utilizando el signo **(|)**.

Nota: Se recomienda consultar el manual para ver las diferentes opciones en el manejo del *Shell*.

La interfaz de usuario Unix/Linux no sólo es a través de *Shell*; sino también de programas llamados utilerías estándar, estos programas se pueden dividir en las siguientes categorías:

1. Comandos para manipular archivos y directorios.
2. Filtros.
3. Herramientas para crear programas (editores y compiladores).
4. Procesadores de texto.
5. Administración del sistema.
6. Diversos.

El estándar POSIX 1003.2 especifica la sintaxis y semántica de aproximadamente 100 programas que pertenecen a las primeras tres categorías y lo que se busca es que cualquier persona pueda escribir scripts de *Shell* y que funcionen en todos los sistemas Unix o Linux. Algunos de los programas utilitarios son:

- cat Concatena varios archivos en la salida estándar.
- chmod Cambia el modo de protección de un archivo.
- cp Copia uno o más archivos.
- ls Produce el listado de un directorio.
- make Compila archivos para construir un binario.
- pr Formatea un archivo para imprimirlo.

- sort Ordena en forma alfabética archivo de líneas.
- Tail Extrae las últimas líneas de un archivo.

Instalación de programas

La instalación de paquetes se explicó en la unidad 4, en el tema 2. El comando **rpm** fue desarrollado por *Red Hat* y se ha convertido en un estándar en todas las distribuciones Linux, inclusive en Debian que utiliza el comando dpkg. Existen distribuciones como Fedora y otras que lo realizan de manera gráfica en la que se van seleccionando las opciones para añadir o eliminar aplicaciones.

Windows NT

En Windows NT el intérprete de comando o *Shell* es similar a Linux; es decir, es una aplicación de consola en la que las aplicaciones de Windows NT interactúan a través de una consola en lugar de elementos GIU, tales como ventanas y cuadros de diálogo. Al iniciar un *Shell* de comandos Windows NT crea una ventana de consola para que se puedan ejecutar los comandos que compartirán la misma ventana de la consola para la salida, la excepción a esto es el comando START que puede crear ventanas adicionales. Para iniciar el *Shell* de comandos por *default*, se tiene que dar click en el botón **Inicio,** posteriormente, **Programas** y, a continuación, seleccionar el comando del **símbolo del sistema**. Otra forma de iniciar el *Shell* es seleccionar **Inicio**, posteriormente, seleccionar **ejecutar** y teclear el comando **cmd** y se iniciará la consola. El *Shell* de comandos se ejecutan de modo interactivo, de esta forma, el *Shell* manda un mensaje y espera a la entrada del teclado, cuando se introduce una línea de comandos, se interpreta inmediatamente y se ejecuta, después de la ejecución, el *Shell* muestra otro sistema y el proceso comienza de nuevo y termina hasta que se ingresa el comando EXIT para terminar la sesión del *Shell*. Al igual que en Linux se pueden programar *scripts* para que los comandos puedan leerse, interpretarse y ejecutarse para un propósito específico. La sintaxis

para ejecutar un comando simple por medio del *Shell* está compuesta por el nombre de un comando seguido de los argumentos necesarios. Por ejemplo. **C:\>copy c:*.back e\ backup /s**, en donde, *copy* es el comando a ejecutar con tres argumentos establecidos y separados por espacios uno de otro.
CMD.EXE y COMMAND.COM

Es importante aclarar que un *Shell* de comando no es un símbolo del sistema MS-DOS aunque compartan el mismo icono. El comando *Shell* de Windows NT es una aplicación de 32 *bits* de la consola que se encuentra en el archivo ejecutable cmd.exe. El símbolo del sistema MS-DOS es una aplicación de DOS de 16 *bits* que se encuentra en el archivo ejecutable COMMAND.COM.

Instalación de programas.

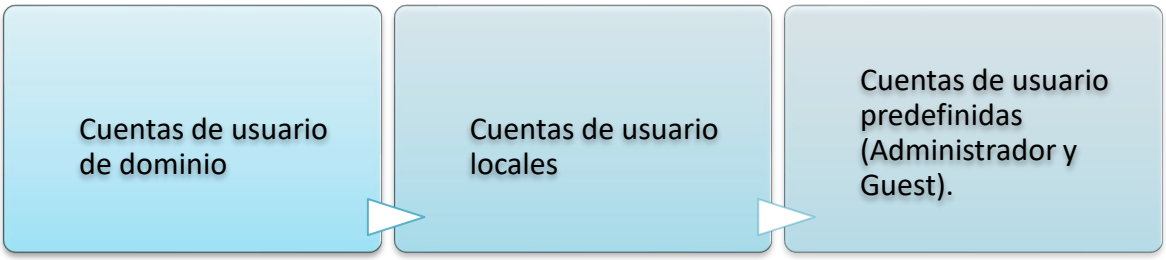
Para la instalación de *software*, Windows NT cuenta con herramientas administrativas como el panel de control que ayudan a instalar o eliminar programas o bien por medio del asistente de instalación de un *software* en particular.

7.8. Creación de usuarios y grupos

En las unidades 3.11 y 4.2 se describen los aspectos importantes para la creación de usuarios y grupos en los sistemas operativos GNU/Linux y FreeBSD, por lo que en este tema se explica sólo lo referente a Windows NT.

Windows NT

En Windows NT una cuenta de usuario proporciona la posibilidad de iniciar una sesión en un dominio para acceder a los recursos de la red o bien iniciar una sesión en una computadora para acceder a los recursos de esa computadora. Windows NT proporciona diferentes tipos de cuentas:



Las cuentas de usuario de dominio permiten a los usuarios iniciar una sesión en el dominio y acceder a los recursos en cualquier parte de la red. El usuario proporciona su nombre de usuario y contraseña, entonces Windows lo autentica y construye un *token* de acceso que contiene información sobre el usuario y los parámetros de seguridad, posteriormente el *token* proporciona el acceso a los recursos durante la duración de la sesión. Cuando se crea una cuenta de usuario de dominio, ésta se replica en el almacén de *Active Directory* sobre un controlador de dominio y el controlador replica la información de la cuenta del nuevo usuario a todos las computadoras que realizan la función de controladores de dominio en un dominio específico.

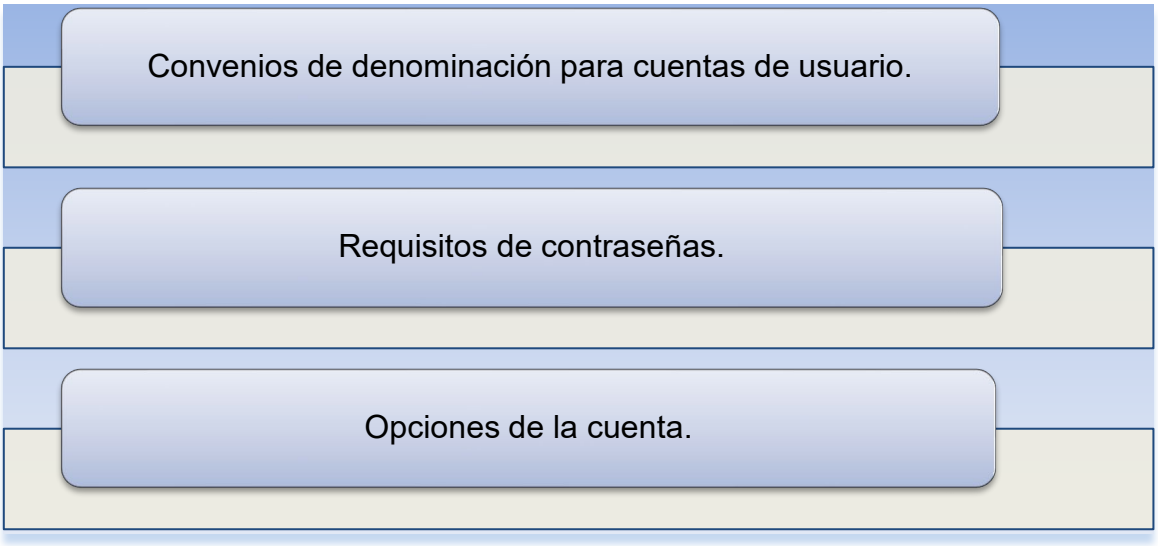
Las cuentas de usuario locales permiten a los usuarios iniciar una sesión y acceder a los recursos sólo en el equipo donde se crea la cuenta de usuario local. Cuando se crea una cuenta de usuario local Windows NT crea la cuenta solamente en la base de datos de seguridad de la computadora y no replica la información de la cuenta local en los controladores de dominio. Después de que se crea la cuenta de usuario local la computadora utiliza la base de datos de seguridad local para autenticar la cuenta de usuario local y le permite iniciar una sesión en la computadora.

Las cuentas predefinidas (Administrador y *Guest*) se utilizan para realizar tareas administrativas. El sistema operativo no permite borrar estas cuentas ni deshabilitar la cuenta Administrador; pero se pueden renombrar.

La cuenta predefinida **Administrador** sirve para administrar toda la computadora y la configuración del dominio y puede crear y modificar las cuentas de usuarios y grupos, administrar las políticas de seguridad, crear impresoras, y asignar permisos y derechos a las cuentas de usuario para que accedan a los recursos locales o remotos. La persona asignada como administrador deberá crear una cuenta de usuario que sirva para tareas no administrativas y utilizar la cuenta administrador sólo cuando se realicen tareas administrativas. La cuenta invitado (*Guest*) se debe utilizar para dar a los usuarios ocasionales la posibilidad de iniciar una sesión y acceder a los recursos. Por ejemplo, un usuario que necesite acceder a los recursos por un corto tiempo puede utilizar la cuenta Invitado (*Guest*).

Planificación de la cuentas de usuario

Con el fin de racionalizar el proceso de la creación de cuentas, se debe planear y organizar la información de los usuarios en tres aspectos:



Los convenios de denominación establecen cómo los usuarios se identifican en un dominio con el fin de que los usuarios recuerden los nombres de inicio de sesión y ubicarlos en una lista, son los siguientes:

- Nombres únicos de inicio de sesión. Los nombres de las cuentas de usuario de dominio deben ser únicas dentro de la OU (Unidad Organizacional) donde se creó la cuenta de usuario de dominio.
- Tamaño de la cuenta. Pueden contener hasta 20 caracteres en mayúscula o minúscula.
- Caracteres no válidos. Los siguientes caracteres no son válidos: **“ / = , + * ¿ [ ] \ < >: ;**
- Caracteres válidos. Se puede utilizar una combinación de caracteres especiales y alfanuméricos para ayudar a identificar las cuentas de usuario. Los nombres de inicio de sesión de usuario no distinguen mayúsculas y minúsculas.
- Nombres duplicados. Si dos usuarios tienen el mismo nombre, se puede utilizar el mismo nombre y la última inicial y agregar letras del apellido para diferenciar los nombres duplicados.

- Tipo de empleo. Es útil identificar de forma temporal a los usuarios por sus cuentas de usuario. Por ejemplo, para identificar temporalmente a los usuarios, se puede utilizar una T y un guion en frente del nombre de inicio de la cuenta: T-Juan.

Los requisitos de contraseñas son para proteger el acceso al dominio o a una computadora, cada cuenta debe tener una contraseña y considerar los siguientes aspectos:

Asignar la contraseña para la cuenta Administrador para evitar el acceso no autorizado a la cuenta.
Determinar si el Administrador o los usuarios controlarán las contraseñas.
Utilizar contraseñas difíciles de adivinar a los ataques de diccionario o fuerza bruta.
Las contraseñas deben tener una longitud mínima de 8 caracteres.
Utilizar letras mayúsculas y minúsculas, números y caracteres no alfanuméricos válidos.

En las opciones de las cuentas, se debe valorar las horas en que un usuario puede iniciar una sesión en la red y las computadoras desde las cuales se puede iniciar ésta, también determinar si las cuentas temporales tienen que expirar y si las cuentas normales de usuario deberían finalizar para configurar la fecha de término para asegurar que la cuenta se deshabilite cuando un usuario no deba tener más acceso a la red o al equipo.

Creación de cuentas de usuario

Para la creación de las cuentas de usuario se debe utilizar el complemento Usuarios y equipos de *Active Directory* (*Active Directory Users And Computers*) para crear una cuenta nueva de usuario de dominio. Cuando se crea una cuenta de usuario de dominio, ésta siempre se crea en el primer controlador de dominio disponible y la cuenta se replica a todos los controladores de dominio. Se debe seleccionar la Unidad Organizacional (OU por sus siglas en inglés) para crear la cuenta. Se puede crear la cuenta de usuario de dominio en la Unidad Organizacional predeterminada Usuarios (*Users*), en una Unidad Organizacional o en las Unidades Organizacionales que se crean para almacenar cuentas de usuarios.

Para la creación de usuarios locales en un solo equipo Windows NT ofrece la utilería Administrador de usuarios. Se encuentra en el Menú Inicio (Programas/Herramientas de administración) y se utilizan la planificación de cuentas de usuario aplicables a las cuentas de usuario de dominio. La siguiente imagen muestra el cuadro de diálogo para crear un usuario local en Windows NT.

Grupos

Un grupo es una colección de cuentas de usuario. Los grupos simplifican la administración de usuarios ya que permiten asignar los permisos y derechos a un grupo de usuarios en lugar de asignarlos a cada cuenta individual. También los usuarios pueden pertenecer a más de un grupo. Cuando se asignan permisos, se permite a los usuarios la capacidad de acceder a los recursos específicos y se define el tipo de acceso que tienen. Ejemplo, si varios usuarios requieren leer el mismo archivo, se agregarán sus cuentas de usuario a un grupo y se le dará al grupo el permiso de leer el archivo. Los derechos permiten a los usuarios realizar tareas del sistema operativo, tales como: cambiar la hora, realizar respaldos de seguridad, restaurar archivos, iniciar una sesión local, etcétera. Además de las

cuentas de usuario se pueden añadir contactos, computadoras y otros grupos a un grupo, lo que simplifica el proceso de asignar una tarea de sistema para que una computadora acceda a un recurso de otra computadora.

Grupos de dominio y locales

Los grupos que se implementan en un dominio se refieren normalmente como grupos, mientras que otros grupos en Windows NT Server se refieren específicamente como grupos locales o grupos predefinidos. Para efectos de este tema, el término grupo se utilizará en el sentido genérico; es decir, a cualquier tipo de grupo. Los grupos se crean por seguridad para asignarles permisos y otras veces por razones no relacionada a la seguridad. Por ejemplo, mandar mensajes de *e-mail*. Para facilitar esto Windows NT incluye dos tipos de grupos:

- Grupos de seguridad. Se utilizan para asignar permiso para acceso a los recursos.
- Grupos de distribución. Se utilizan cuando la única función del grupo no está relacionada con la seguridad y no se pueden asignar permisos.

Cuando se crea un grupo se debe seleccionar un tipo de grupo y un ámbito de grupo. Los ámbitos de grupo permiten utilizar los grupos de forma diferente para asignar permisos y determinan donde se puede utilizar el grupo en la red. Los ámbitos de grupo son:

- ✓ Grupos locales de dominio. Se utilizan para asignar permisos a los recursos. Su característica principal es que se pueden agregar miembros desde cualquier dominio (Permanencia abierta) y asignar permisos para acceder solamente a los recursos que se ubican en el mismo dominio donde se ha creado el grupo local de dominio (Acceso a recursos en un dominio).
- ✓ Grupos globales. Se utilizan para organizar los usuarios que comparten requisitos de acceso similares a la red. Sus características son:

- ✓ Permanencia limitada. Permiten añadir miembros solamente desde el dominio en el cual se ha creado el grupo global.
- ✓ Acceso a recursos en cualquier ámbito. El grupo global se puede utilizar para asignar permisos para acceder a los recursos que están ubicados en cualquier dominio.
- ✓ Grupos universales. Se utilizan para asignar permisos a los recursos relacionados en varios dominios. Sus características son:
  - ✓ Pertenecía abierta. Se pueden agregar miembros desde cualquier dominio.
- ✓ Acceder recursos en cualquier dominio. Se puede utilizar para asignar permisos para acceder a recursos situados en cualquier dominio.

Creación de grupos en un dominio.

Para la creación y eliminación de grupos de dominio se debe utilizar el complemento Usuarios y equipos de *Active Directory* (*Active Directory Users And Computers*). Los grupos deben crearse en la OU Usuarios (*Users*) o en una OU que se haya creado específicamente para los grupos. Es importante mencionar que también existen los grupos locales de dominio que se crean en el almacén de *Active Directory* y los utilizan todos los controladores de dominio en un dominio específico.

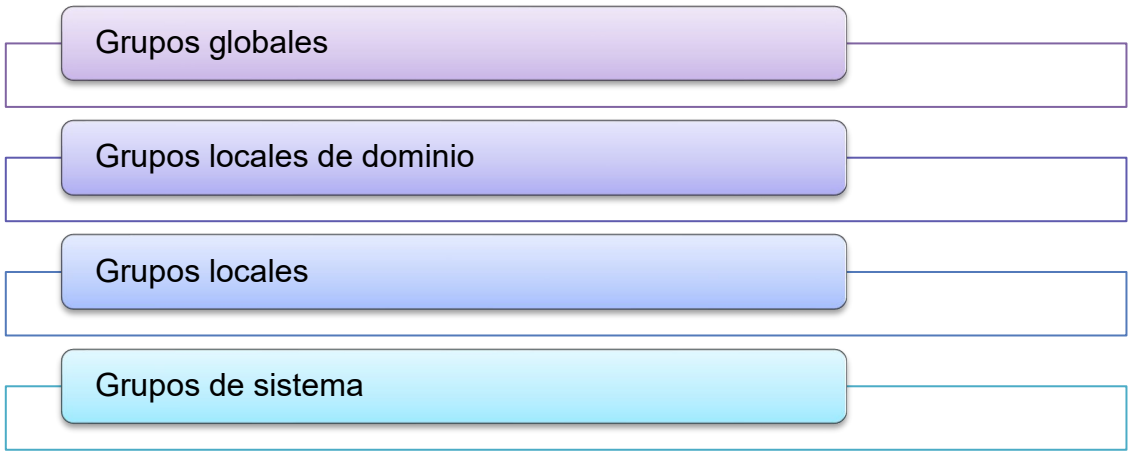
Creación de grupos locales

Los grupos locales, que no son de dominio, se crean en servidores independientes, servidores miembros del dominio y en equipos con Windows NT Server. Estos grupos sólo se pueden utilizar en la computadora en donde se creó el grupo local. Un grupo local puede tener cuentas de usuario en el equipo y se pueden asignar recursos a ese equipo. Para la creación de grupos locales se debe utilizar el complemento Administración del equipo (*Computer Management*). Se crean los grupos locales en el directorio o carpeta Grupos (*Groups*). Para crear un grupo local

hay que expandir Usuarios locales y grupos (*Local Users And Groups*) en el árbol de consola y seleccionar Grupos (*Groups*) del menú Acción (*Action*) y entonces pulsar en Grupo nuevo (*New Group*) e introducir el nombre y descripción del grupo.

Grupos predefinidos

En las versiones actuales de Windows NT Server existen diversos tipos de grupos para que no se tengan que crear y asignar permisos para funciones que son comúnmente utilizadas. Los grupos son los siguientes:



Grupos globales. Cuando se crea un dominio se crean grupos globales predefinidos en el almacén de *Active Directory*. Se asignan derechos agregando los grupos globales a grupos locales de dominio o explícitamente asignando derechos de usuario o permisos a los usuarios globales predefinidos. La OU (Unidad Organizacional) Usuarios (*Users*) contiene los grupos globales predefinidos en un dominio. Los grupos globales de dominio son los siguientes:

- ✓ Usuarios de dominio (*Domain Users*). Se agregan automáticamente Usuarios de dominio (*Domain Users*) al grupo local predefinido Usuarios (*Users*). La cuenta Administrador (*Administrator*) es inicialmente un miembro

y Windows NT Server hace automáticamente miembro a cada cuenta nueva de usuario de dominio.

- ✓ Administrador de dominio (*Domain Admins*). Se agregan automáticamente Administradores de dominio (*Domains Admins*) al grupo local de dominio Administradores (*Administrators*) para que los miembros de Administradores de dominio (Domain Admins) puedan realizar tareas administrativas en cualquier computadora en el dominio.

- ✓ Invitados de dominio (*Domain Guests*). La cuenta de invitado (*Guests*) es un miembro del dominio, la cuenta está deshabilitada

- ✓ Administradores de empresa (*Entreprise Admins*). Se pueden agregar cuentas de usuario a Administradores de empresa (*Enterprise Admins*) para usuarios que deben de tener el control administrativo sobre toda la red. La cuenta se le considera un miembro del dominio.

Grupos locales de dominio. Windows NT Server crea grupos locales de dominio predefinidos en un dominio, para proporcionar a los usuarios los derechos de usuario y permisos para realizar actividades en los controladores de dominio y en el almacén de *Active Directory*. Un grupo local predefinido realiza las mismas funciones que un grupo local de dominio, con la diferencia de que un grupo local predefinido no se puede borrar. Los grupos que contiene son:

- ✓ Operadores de cuenta (*Account Operators*). Estos grupos pueden crear, borrar y modificar cuentas de usuario y grupos; los miembros no pueden modificar el grupo Administradores.

- ✓ Operadores de servidores (*Server Operators*). Pueden compartir recursos de disco y realizar copias de seguridad, así como restaurar archivos en un controlador de dominio.

- ✓ Operadores de impresión (*Print Operators*). Pueden configurar y gestionar las impresoras de la red en los controladores de dominio.

- ✓ Administradores. Pueden realizar tareas administrativas en todos los controladores de dominio y en el dominio específico, son miembros la cuenta de usuario, Administrador, el grupo global Administradores del dominio, el grupo global Administradores del dominio y el grupo global Administradores de empresa.
- ✓ Invitados (*Guests*). Pueden realizar sólo tareas para las cuales se les haya concedido derechos y acceder solamente a los recursos para los cuales se asignaron permisos. De forma predeterminada son miembros de la cuenta Invitados (*Guest*) y el grupo global de Invitados de dominio (*Domain Guests*).

- ✓ Operadores de copia (*Backup Operators*). Los miembros pueden realizar copias de seguridad y restaurar todos los controladores de dominio por medio de *Windows Backup*.

- ✓ Usuarios (*Users*). Los miembros pueden realizar tareas para los cuales tengan derechos y acceder solamente a los recursos que tienen los permisos asignados. Son miembros el grupo Usuarios de dominio (*Domain Users*), el grupo especial Usuarios autentificados (*Authenticated Users*) y el grupo especial *INTERACTIVE*.

Grupos locales predefinidos. Todos los servidores independientes, servidores miembros del dominio y computadoras con Windows NT Server tienen grupos locales predefinidos. Estos grupos dan derecho a realizar las tareas de sistema en una única computadora, tales como cambiar la hora y administrar los recursos del

sistema. Se ubican en la carpeta Grupos (*Groups*) en el complemento Administración del equipo (*Computer Management*); al igual que los grupos predefinidos en el dominio, no se pueden borrar los grupos locales que no corresponden al dominio predefinido. Los grupos locales predefinidos son:

- ✓ Usuarios (*Users*). Los miembros pueden realizar solamente tareas y acceso a los recursos para los cuales se han asignado permisos. De forma predeterminada Windows NT agrega las cuentas de usuario local que se han creado en la computadora en el grupo Usuarios (*Users*).

- ✓ Administradores (*Administrators*). Pueden realizar tareas administrativas en el equipo. De forma predeterminada la cuenta de usuario predefinida Administrador (*Administrator*) del equipo es un miembro.

- ✓ Invitados (*Guests*). Pueden realizar tareas y acceder a los recursos con los permisos asignados, no pueden realizar cambios permanentes en su entorno de escritorio. La cuenta está deshabilitada durante la instalación.

- ✓ Operadores de copia (*Backup Operators*). Pueden utilizar Windows Backup para realizar copias de seguridad y restaurar un equipo.

- ✓ Usuarios avanzados (*Power Users*). Estos miembros pueden modificar las cuentas de usuario locales en la computadora y compartir recursos.

- ✓ Duplicadores (*Replicator*). Los miembros pueden usar y configurar los servicios de réplica de archivos.

Grupos de sistema predefinidos. En Windows NT Server también incluye grupos especiales que se utilizan sólo para mostrar cómo un usuario particular está utilizando el sistema Windows NT en un momento dado. Estos grupos no tienen

miembros reales, el número de miembros de un grupo viene dado por la forma como acceden los usuarios a un recurso. Los grupos genéricos son:

- ✓ *Interactive*. Cualquier usuario que ha iniciado localmente una sesión.
- ✓ *Network*. Cualquier usuario que ha conseguido acceder a un recurso a través de la red.
- ✓ *System*. El sistema operativo.
- ✓ *Creator*/*Owner*. Quien esté creando un archivo, subdirectorio o una cola de impresión.

RESUMEN DE LA UNIDAD

Cuando se ha determinado el tipo de sistema operativo que se utilizará, sigue la etapa de la implantación, ésta deberá planearse considerando aspectos como los requerimientos de *hardware*, compatibilidad con otros elementos de *hardware* y *software*, sistema de archivos, aplicaciones, espacio del disco, usuarios, etcétera. Uno de los aspectos más importantes durante la implantación y administración de un equipo, es entender que la cuenta maestra del administrador conocida como superusuario, *root* o administrador, no deberá utilizarse para realizar tareas cotidianas como enviar o recibir e-mail, programación, etcétera para esto se deben crear cuentas de usuario normales en las que, si existe algún error, no afecte al sistema. En la selección de un sistema operativo, siempre se debe considerar la función que realizará el servidor, las características del sistema operativo, tipo de licencias, etcétera. La manera más sencilla de crear un disco de arranque de Linux es descargando la imagen de la distribución deseada desde el siguiente enlace: http://www.linux.org/dist/list.html. En Windows NT al ser un sistema comercial, deberán adquirirse el tipo de licencias a instalar, los discos que se entregan contienen el sistema operativo, por lo que no se requiere la creación de discos de arranque. Una de las actividades más críticas es determinar el espacio que tendrán los discos duros, por lo que se requiere de una planeación que considere aspectos como la función que tendrá el servidor, tipo y versión o distribución del sistema operativo, sistema de archivos, crecimiento, etcétera. La asignación del espacio en el disco o discos duros se realiza por medio del proceso denominado “partición” que consiste en dividir un disco físico en varios discos lógicos, éste se puede realizar de manera manual o automática, existen dos tipos de particiones: primarias y extendidas.

En Windows, las particiones reconocidas son identificadas con una letra seguida por un signo de doble punto (p. e. C:\). En sistemas basados en Linux, se le asigna un archivo especial en el directorio /dev a cada partición (p. e. hda1, sda2, etcétera); el archivo recibe un nombre compuesto de tres letras seguidas de un número. La creación del sistema de archivos proporciona el mecanismo para el almacenamiento y el acceso en línea de los datos y programas de un sistema operativo en particular y usuarios del mismo, de manera general, se deben realizar las siguientes acciones: Crear el sistema de archivos, formatear la partición creada y montar el sistema de archivos. Para la gestión del espacio libre el sistema operativo utiliza mecanismos internos (Mapa de *bits*, lista enlazada, conteo, etcétera) y herramientas o comandos (du, df. compresión de datos, comprobar disco, etcétera). El *Shell* es una interfaz entre el usuario y un sistema operativo en particular, se ejecutan los comandos en un ambiente de consola y el *Shell* los interpreta y ejecuta, sin hacer uso de los entornos gráficos (GUI). En Windows NT una cuenta de usuario permite iniciar una sesión en un dominio para acceder a los recursos de la red o bien iniciar una sesión en una computadora para acceder a los recursos sólo de ese equipo; es decir, de manera local. La cuenta **Administrador** permite administrar toda la computadora y la configuración del dominio y puede crear y modificar las cuentas de usuarios y grupos, administrar las políticas de seguridad, crear impresoras, y asignar permisos y derechos a las cuentas de usuario para que accedan a los recursos locales o remotos. La cuenta invitado (*Guest*) permite proporcionar a los usuarios ocasionales la posibilidad de iniciar una sesión y acceder a los recursos de manera temporal. También existen diferentes opciones para la creación de cuentas y grupos locales o de dominio.

BIBLIOGRAFÍA

SUGERIDA

Autor	Capítulo	Páginas
Flynn, Ida	11	268-270
Silberschatz, Abraham	11	386-388
Tanenbaum, Andrew	6	683-688
Carretero, Jesús	11 12	613-614 620-647
Microsoft Corporation	2 4 7	53-59 126-131 314-330

UNIDAD 8

TÓPICOS AVANZADOS DE SISTEMAS OPERATIVOS

SUAyED UNAM
Sistema Universidad Abierta y Educación a Distancia

Plan 2012 2016 actualizado

OBJETIVO PARTICULAR

El alumno utilizará los conceptos de eficiencia y rendimiento de los sistemas operativos, la importancia de la incorporación de los controladores y las características de los sistemas operativos de red y distribuidos.

TEMARIO DETALLADO

(6 horas)

8. Tópicos avanzados de sistemas operativos

8.1. Eficiencia y rendimiento o desempeño del SO
8.2. Escritura de *drivers*
8.3. Sistemas operativos de red
8.4. Sistemas operativos distribuidos
8.5. Servicios remotos en internet

INTRODUCCIÓN

La eficiencia y rendimiento de los sistemas operativos depende de muchos factores; sin embargo, la lentitud de muchos de éstos se debe principalmente al propio sistema operativo. Por otro lado, la industria de la computación, la informática, la electrónica y las telecomunicaciones han tenido un avance tecnológico considerable, logrando sistemas con mayores capacidades como los sistemas distribuidos y de red que están basados en procesadores y componentes heterogéneos u homogéneos conectados a través de redes de altas velocidades o de internet utilizando diversos protocolos y servicios de red, lo que permite aumentar sus capacidades de cómputo, procesamiento y aplicaciones en todo el mundo. En la presente unidad se describe la importancia de medir el rendimiento en los sistemas operativos, su monitorización y la forma de mejorarlo. La arquitectura y el *software* de E/S y su relación con los drivers o controladores. También se explican las características y funciones de los sistemas operativos de red y distribuidos, así como los servicios y protocolos más importantes de internet que hacen posible la comunicación entre distintos sistemas.

8.1. Eficiencia y rendimiento o desempeño del SO

En ambientes similares, un sistema operativo rápido es mejor que uno lento; sin embargo, un sistema puede ser rápido y poco confiable, en contraste un sistema puede ser lento, pero confiable. Las optimizaciones complejas pueden causar errores de diverso tipo y deben aplicarse sólo si en realidad se requieren, aunque hay puntos en el que es importante el desempeño, por lo que éste deberá optimizarse. La lentitud de muchos sistemas operativos se debe principalmente al diseño propio de los mismos. Por ejemplo, los sistemas antiguos como MS-DOS y la versión 7 de *Unix*, requerían unos cuantos segundos para arrancar. Los sistemas modernos como Unix o Windows NT que contienen *hardware* mucho más poderoso, pueden tardar minutos en hacerlo. Esto se debe a que realizan un mayor trabajo se requiera o no, como levantar diversos *drivers*, exploración completa del *hardware*, aplicaciones más complejas, etcétera. Son muchos los factores que permiten mejorar el desempeño de un sistema operativo; en primer lugar, son los diseñadores que añaden y mejoran nuevas funciones, otro factor es la mercadotecnia de los sistemas, ya que cuando sale al mercado una nueva versión, lo más seguro es que ya se hayan incluido todas las mejoras y funciones que de verdad sean útiles para el usuario. La adición de nuevas versiones con nuevas funciones que no se requieren, posiblemente mejoren la venta de los productos, pero no mejorarán el desempeño. Sin embargo, el administrador de un equipo puede medir y mejorar el desempeño de un sistema operativo para mejorar su rendimiento.

Medición del rendimiento.

El autor Flynn define el rendimiento como “la eficiencia con la cual un sistema de cómputo llena sus metas” (Flynn, 2001: 270-274); es decir, que da un buen servicio a sus usuarios. Medir la eficiencia de un sistema no es fácil, ya que se ve afectada por tres componentes principales: los programas del usuario, los programas del sistema operativo y las unidades del *hardware*. También, el rendimiento del sistema puede ser muy subjetivo y difícil de cuantificar; por ejemplo, ¿cómo medir la facilidad de uso? Aunque existen factores que sí pueden cuantificarse más objetivamente como el número de accesos al disco por minuto, aunque en este caso, esta medida no es absoluta, sino relativa, pues está basada en interacciones de los tres componentes y de la carga de trabajo manejada por el sistema operativo. Los diseñadores, analistas y administradores de equipos se apoyan en las siguientes mediciones del rendimiento:

- Producción. Es una medida que indica la productividad de un sistema como un total y se mide en condiciones de carga estática; número de tareas procesadas por día o el número de transacciones en línea manejadas por hora. También puede medirse como un volumen de trabajo manejado por una unidad específica del sistema de cómputo que resulta útil para encontrar cuellos de botella.
- Capacidad. Los cuellos de botella se generan cuando el uso de los recursos llega a su capacidad; es decir, a su máximo nivel de producción, por lo que los recursos se saturan y los procesos no se ejecutan, la hiperpaginación y la memoria son los elementos más comunes que llegan a saturarse. La capacidad y la producción pueden monitorizarse con *hardware* o *software* específico, por lo que pueden detectarse a tiempo para tomar una acción apropiada y resolver el problema.

- Tiempo de respuesta. Es una medida muy importante que considera el “tiempo” para medir el rendimiento del sistema, el tiempo es el intervalo requerido para procesar una solicitud del usuario desde el momento en que oprime la tecla para enviar el mensaje hasta que el sistema indica la recepción del mismo. Esta prueba se aplica para procesamiento en línea (usuarios del sistema).
- Prueba de retorno. Es una prueba que considera un tiempo desde el momento en que se envía la tarea, hasta que su salida vuelve al usuario “tiempo de retorno”. Esta medida depende de la carga de trabajo manejada por el sistema y el tipo de tarea que se ha enviado, ya que algunas solicitudes se manejan con mayor rapidez que otras por que requieren menos recursos. Esta prueba se aplica al procesamiento por lotes. Para que esta prueba tenga una medida más precisa de la predictibilidad del sistema, el tiempo de respuesta y el tiempo de retorno deben incluir los valores estadísticos tanto del promedio como de la varianza.
- Utilización de recursos. Es una medida de cuanto contribuye cada unidad del sistema a la operación general del mismo. Por lo general, se da como un porcentaje del tiempo que un recurso está en uso. Por ejemplo, el CPU está ocupado 50%, la impresora en red 80%, etcétera, este tipo de datos ayudan al administrador a determinar si existe un equilibrio entre las unidades del sistema, o si se encuentran limitados por entradas y salidas o por el CPU.
- Disponibilidad. Esta prueba indica la probabilidad de que un recurso, esté disponible cuando lo requiera el usuario. Para los usuarios en línea, puede significar la probabilidad de que un puerto o terminal estén libres cuando se requieran. Para los usuarios que ya están en el sistema, puede denotar la probabilidad de que uno o varios recursos (impresora, cintas, discos, etcétera) estén listos cuando el programa realice la solicitud; es decir, la disponibilidad significa que una unidad puede operar y no estar fuera de servicio cuando un usuario lo necesita. Los factores para obtener la disponibilidad son: Tiempo promedio entre fallas (MTBF) y Tiempo medio

para reparar (MTTR). El MTBF es el tiempo promedio que una unidad está en operación antes de que tenga alguna interrupción por alguna falla. El MTTR es el tiempo promedio necesario para reparar una unidad que presentó una falla y que, una vez reparada, se pondrá en servicio; su fórmula es: A= MTBF/MTBF- MTTR.

Monitorización

Como se mencionó anteriormente, los sistemas computacionales han evolucionado tecnológicamente y, aunado a esto, se han desarrollado varias técnicas de monitorización para medir su rendimiento, las cuales pueden aplicarse por medio de *hardware* o *software*. Los monitores de *hardware* son más costosos; pero tiene la ventaja de tener poco impacto en el desempeño del sistema, ya que se instalan en el exterior del equipo y se conectan de manera electrónica. Estos incluyen: contadores, relojes, y elementos comparativos conectados electrónicamente. Los monitores basados en *software* son generalmente menos costosos; pero pueden llegar a distorsionar los resultados del análisis, porque su instalación pasa a formar parte del sistema; es decir, utiliza y revisa los recursos propios del sistema con el software de monitoreo al mismo tiempo**.** También se tiene que desarrollar un software de monitoreo para cada sistema en particular y es difícil trasladarlo de un sistema a otro. Actualmente, las mediciones del sistema incluyen no sólo la velocidad del CPU; sino también otros elementos como son: unidades de *hardware*, sistemas operativos, compiladores y diversos *software* del sistema. Las mediciones se pueden realizar de diversas maneras, algunas utilizan programas reales; es decir, *software* que está en producción en ese momento y originan resultados que se conocen como marcas de referencia (*benchmarks*) que son útiles para comparar sistemas que han sufrido cambios considerables. Los fabricantes constantemente utilizan las marcas de referencia para demostrar a sus potenciales clientes las ventajas de un nuevo modelo de CPU, sistema operativo, compilador o unidad de *hardware*, etcétera.

Optimización

Existen algunas formas que permiten mejorar el desempeño de un sistema operativo:

- Equilibrio espacio-tiempo. Consiste en un método general para mejorar el desempeño de un sistema operativo sacrificando tiempo a cambio de espacio. En computación hay que tomar decisiones entre un algoritmo que consume poca memoria, pero es lento y uno que consume mucha memoria y es más rápido; es decir, al realizar una optimización importante, es conveniente buscar algoritmos que sean rápidos ocupando más memoria o, bien, que ahorren memoria realizando más cálculos. Una técnica útil consiste en sustituir procedimientos pequeños por macros. El uso de una macro elimina el proceso adicional que normalmente conlleva una llamada a procedimientos. La ganancia es más importante si la llamada se realiza dentro de un ciclo.

- Uso de *cachés*. Es una técnica que permite mejorar el uso de *cachés*. Puede aplicarse en situaciones en donde es posible que se vaya a necesitar el resultado varias veces; es decir, se puede realizar todo el trabajo la primera vez y luego almacenar el resultado en un *caché*. Posteriormente, se verifica el *caché*, si el resultado está ahí, se utiliza; si no, se vuelve a realizar de nuevo todo el trabajo.

- Sugerencias. Las entradas de *caché* siempre son correctas. Una búsqueda en *caché* puede fallar; pero si encuentra una entrada, se garantiza que es correcta y puede utilizarse. En algunos sistemas es conveniente tener una tabla de “sugerencias”. Éstas son sugerencias en cuanto a la solución, y no se garantiza que sean correctas. El invocador debe verificar el resultado por

su cuenta. Por ejemplo, una sugerencia son las URL incrustadas en las páginas *Web*. Hacer *click* en un vínculo, no garantiza que la página *Web* a la que apunte esté ahí, ya que pudo ser retirada años atrás. Por lo que la información en la página que apunta es una sugerencia.

- Localidad. Los procesos y programas no actúan al azar, muestran un alto grado de localidad en el tiempo y espacio y se puede aprovechar esta información para mejorar el desempeño. Por ejemplo, las páginas que un proceso está utilizando en forma activa pueden tomarse como su conjunto de trabajo, y el sistema operativo se asegura de que, cuando se permita ejecutar el proceso, el conjunto de trabajo esté en la memoria, lo que reducirá el número de fallas de página. Este principio también se aplica para los archivos. Una vez que un proceso selecciona su directorio, es probable que varias de sus referencias futuras sean a archivos que están en ese directorio. Si se colocan todos los nodos-i y archivos cercanos entre sí en el disco, podría mejorarse el desempeño. Este principio es a base del *Fast File System* de Berkeley (Sistema operativo derivado de Unix nacido de los aportes de la Universidad de California de Berkeley). Otra área en la que la localidad desempeña un papel importante es la calendarización de subprocesos en multiprocesadores que consiste en tratar de ejecutar cada subproceso en el último CPU que uso, con la idea de que algunos de sus bloques de memoria todavía se encuentren en el *caché* de la memoria (Tanenbaum, 2003: 883-889).

8.2. Escritura de drivers

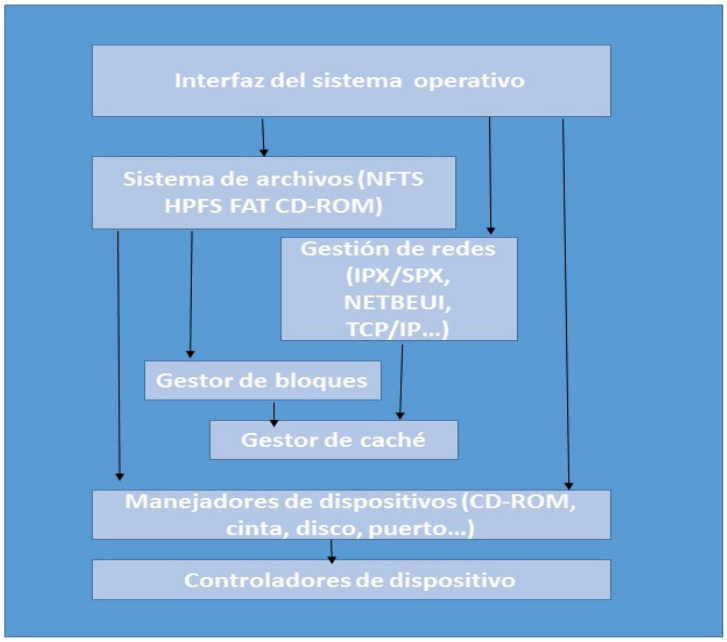
Las operaciones de E/S y el procesamiento son las dos tareas fundamentales de una computadora y en muchos casos, las operaciones de E/S es la tarea principal. La función del sistema operativo en el sistema de E/S es administrar y controlar las operaciones y los dispositivos de E/S. Este sistema está construido como un conjunto de manejadores o *drives* apilados y cada uno está asociado a un dispositivo de entrada/salida (archivos, red, etcétera). Un *driver* o controlador de dispositivo es un componente de *software* que utiliza el sistema operativo para interactuar con el *hardware;* por ejemplo, un monitor o impresora. Aunque no todos los *drivers* son de dispositivos; por ejemplo, el *driver* de *software* para un sistema de archivos como ext3 o *ReiserFS* que permiten mapear las estructuras de datos de bajo nivel a estructuras de alto nivel. El *driver* está ubicado entre el *hardware* y la aplicación. La importancia de programar un *driver* es principalmente para dar soporte a un nuevo *hardware* o *software* y mantener productos propios.

Para comprender su función, es importante describir la arquitectura del sistema de entrada/salida y posteriormente la importancia del *software* de E/S.
La arquitectura del sistema de E/S está formada en capas y cada capa tiene una función definida:

- Interfaz del sistema operativo para E/S. Proporciona los servicios de E/S síncrona y asíncrona para las aplicaciones y una interfaz homogénea para comunicarse con los *drives* de los dispositivos.
- Sistema de archivos. Proporciona una interfaz homogénea a través del sistema de archivos virtuales para acceder a todos los sistemas de archivos que proporciona el sistema operativo (FFS, NTFS, FAT, etcétera).

- Gestor de redes. Proporciona una interfaz homogénea para acceder a todos los sistemas de red que proporciona el sistema operativo (TCP/IP, Novell, etcétera).
- Gestor de bloques. Permite operaciones al nivel denominado “bloque” (operaciones que entiende el dispositivo) e interactúa con el *caché* de bloques para optimizar la E/S.
- Gestor de *caché*. Optimiza la E/S por medio de la gestión del almacenamiento intermedio en memoria para dispositivos de E/S de tipo bloque.
- Manejadores de dispositivo o *drivers*. Proporcionan operaciones de alto nivel sobre los dispositivos, y éstos las traducen en su ámbito interno a operaciones de control de cada dispositivo específico Los *drivers* se comunican con los dispositivos reales mediante puertos o zonas de memoria especiales (Carretero: 363-369).

En el siguiente esquema se puede ver una muestra de la arquitectura de entrada/salida:



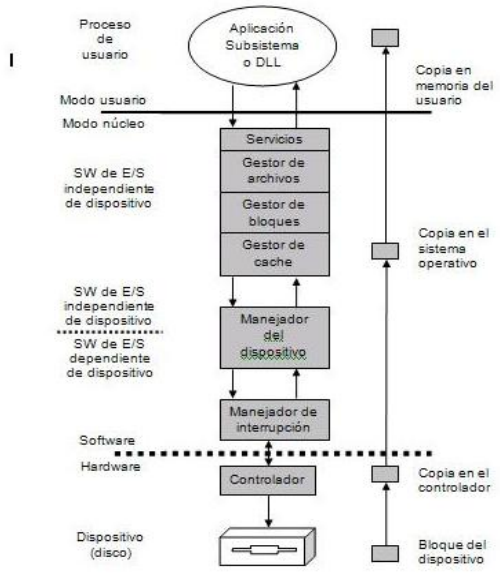
Arquitectura del sistema de entrada/salida, (Carretero, 2001: 365)

Software de E/S

Una vez descrita la arquitectura de E/S de una computadora y las técnicas posibles de transferencia entre el procesador y los periféricos se explica la forma en la que estructura el sistema operativo el *software* de gestión de E/S. El *software* se organiza en capas que corresponden en general con los niveles de la arquitectura de E/S.

En la siguiente figura se muestran los procesos del usuario que realizan peticiones de entrada/salida al sistema operativo. Cuando un proceso solicita una operación de E/S el sistema prepara la operación y bloquea el proceso hasta que se recibe una interrupción del controlador del dispositivo indicando que la operación está completa.

En el siguiente esquema se puede ver une ejemplo de estructuración del *software* de E/S y flujo de una operación de E/S.



Estructura del software de E/S y el flujo de una operación de E/S
(Carretero, 2001: 366)

Las peticiones se procesan de forma estructurada en las siguientes capas:

- Manejadores de interrupción. Tratan las interrupciones que generan los controladores de dispositivos, cuando están listos para la transferencia de datos o bien cuando han leído o escrito los datos de memoria principal en caso de acceso directo a memoria.

- Manejadores de dispositivos o *drivers*. Cada dispositivo de E/S, o cada clase de dispositivos, tiene un *driver* asociado en el sistema operativo. Dicho manejador incluye: código independiente del dispositivo para proporcionar al nivel superior del sistema operativo una interfaz de alto nivel y el código dependiente del dispositivo necesario para programar el controlador del dispositivo a través de sus registros y datos. En los sistemas modernos como Windows NT los *drivers* se agrupan en clases. Para cada clase existe uno genérico que realiza las operaciones de E/S para una clase de dispositivos como: CD-ROM, disco, cinta, teclado, etcétera. Cuando se instala un dispositivo específico, por ejemplo, un disco de una marca y modelo específico, se crea la instalación del *driver* de clase con los parámetros específicos de ese objeto. Dentro del *kernel* se encuentran los *drivers* de dispositivos de bloques y dentro de este conjunto de código, se pueden diferenciar dos niveles: uno, donde se encuentra un *software* independiente del *hardware* y, otro, donde se encuentran los *drivers* propios de cada dispositivo.
- *Software* de E/S independiente de los dispositivos. Está formado por la parte de alto nivel de los manejadores, el gestor de *caché*, el gestor de bloques y el servidor de archivos. La principal función de esta capa de *software,* es ejecutar las funciones de E/S que son comunes a todos los dispositivos a través de una interfaz uniforme.
- Interfaz del sistema operativo. Llamadas al sistema que utilizan las aplicaciones del usuario.

8.3. Sistemas operativos de red

Un sistema operativo de red (NOS) es una computadora conocida como servidor que proporciona servicios a las estaciones de trabajo conectadas en red conocidas como clientes. Muchos sistemas operativos de red modernos realizan las cuatro funciones:

- Administrativas.
- Administración de la memoria.
- Planificación de procesos.
- Administración de los archivos y administración de los dispositivos que incluye las operaciones de disco y de E/S.

Estos sistemas, además, incluyen varias funciones de administración de la red como: las comunicaciones, protocolos, servicios, etcétera, estas funciones se activan únicamente cuando el sistema requiere utilizar la red, en caso contrario, se mantiene en estado latente y el sistema operativo funciona como sistema independiente. El enfoque general de un NOS es compartir los recursos en vez de ejecutar aplicaciones; es decir, dejan a las estaciones de trabajo la capacidad de compartir los recursos del servidor (aplicaciones, periféricos, módems, etcétera). Windows NT Server, Unix, *Netware*, Ubuntu Linux y GNU/Linux son algunos ejemplos de sistemas operativos de red (Flynn, 2000: 252-255). Tienen las características y funciones principales que se expondrán más adelante.

Características de NOS

- En su mayor parte, están implementados por *software* de 16, 32 o 64 *bits*.
- El *software* de 32 *bits* aprovecha toda la capacidad de los procesadores modernos.
- Las redes pueden ser heterogéneas al implementar estaciones de diferentes sistemas operativos (MS-DOS, Unix, Windows, Macintosh, etcétera).
- Pueden operar una amplia variedad de aplicaciones de *software* desarrollado por terceros, así como *hardware* diversos (discos, módems, *switches*, CD-ROM, interfaces de red, etcétera).
- Soporta *software* multiusuario.
- Soporta una amplia gama de protocolos de red (TCP-IP, NetBEUI, etcétera).
- Son sistemas seguros en ambientes de red, combinados con los mecanismos de la seguridad informática.
- Soporta diferentes medios de transmisión y estándares para redes de área local (LAN) y redes de área amplia (WAN).

Funciones de NOS

- Permitir a los usuarios autorizados el acceso al *hardware* y al *software* en un sitio remoto.
- Manejo de comandos de comunicación:
  - ✓ Emulación de terminal remota asíncrona
  - ✓ TCP/IP
  - ✓ Telnet
  - ✓ SMTP (Protocolo Simple de Transferencia de correo)
  - ✓ SNMP (Protocolo Simple Manejador de Red)
  - ✓ SNA (Comunicaciones punto a punto)
  - ✓ APPC (Comunicación Avanzada Programa a Programa)
  - ✓ FTP (Protocolo para Transferencia de Archivos)

- ✓ NetBIOS (Protocolo para compartir recursos)
- ✓ NetBEUI (Protocolo e interfaz para PC-IBM)
- ✓ Otros

8.4. Sistemas operativos distribuidos

El crecimiento de los sistemas distribuidos, Internet y la *World Wide Web* han influido notablemente en el desarrollo de los sistemas operativos y es a mediados de la década de 1990 cuando la conectividad a redes pasó a ser un elemento esencial en un sistema de cómputo. Hoy prácticamente cualquier computadora o dispositivo cuenta con navegadores y protocolos de comunicación que les permite conectarse en red por medio de las redes locales, líneas telefónicas, enlaces dedicados, etcétera. Un sistema operativo distribuido es un conjunto de procesadores que no comparten memoria ni reloj; es decir, cada procesador tiene su propia memoria local, y los CPU se comunican entre ellos mediante diversos esquemas de comunicación, mencionados anteriormente. El propósito de un sistema distribuido es proporcionar un ambiente eficiente y conveniente para compartir los recursos. Los procesadores en un sistema distribuido pueden variar en tamaño y función, desde pequeñas computadoras hasta grandes sistemas de cómputo; así mismo, un sistema distribuido proporciona a sus usuarios el acceso a los múltiples recursos con que cuenta. El acceso a un recurso compartido, permite una mayor velocidad de procesamiento, disponibilidad y confiabilidad de datos (Silberschatz, 2002: 469-472).

Características.

- Compartimiento de recursos. Si varios sitios están conectados entre sí, y cada sitio tiene diferentes capacidades, los usuarios pueden acceder a los recursos que ofrecen (*hardware, software*, datos). Por ejemplo, el usuario A puede utilizar una impresora en un sitio, mientras que el usuario B puede acceder a un archivo que resida en otro sitio.
- Rendimiento. El uso de múltiples procesadores permite la construcción de un sistema de alta capacidad y velocidad de procesamiento de forma simultánea. Si un sitio está sobrecargado, ciertas tareas pueden transferirse a otros sitios para lograr una carga más ligera.
- Confiabilidad. En caso de falla en un sitio, los sitios restantes pueden continuar funcionando, lo que proporciona al sistema una gran confiabilidad. Si el sistema se compone de instalaciones autónomas (computadoras de propósito general), la falla de una de ellas no debe afectar a las demás, si el sistema está compuesto de equipos pequeños y cada uno es responsable de ciertas funciones, (E/S, sistema de archivos, etcétera) entonces, sí puede detener la operación de todo el sistema. Para solucionar esto, deberán existir mecanismos de redundancia en *hardware*, datos y comunicaciones. La falla de algún sitio debe ser detectado por el sistema y sus servicios no deben ser utilizados mientras esté caído, por lo que su función debe ser asumida por otro sitio. Cuando el sitio que falló se recupera deben existir mecanismos que lo integren de nuevo al sistema de forma transparente y armoniosa.

El siguiente cuadro muestra las diferencias importantes entre los sistemas operativos de red (NOS) y los sistemas distribuidos:

Sistema Operativo de red (NOS)	Sistema Operativo distribuido (DO/S)
Recursos propiedad de los sitios locales	Recursos propiedad del sistema global
Recursos locales administrados por el sistema operativo local	Recursos locales administrados por un DO/S global
Acceso ejecutado mediante un sistema operativo local	Acceso ejecutado por el DO/S
Solicitudes pasadas de un sistema operativo local a otro vía el NOS	Solicitudes pasadas directamente de un sitio a otro vía el DO/S

8.5. Servicios remotos en Internet

Internet es una red de redes de alcance mundial que utiliza la suite de protocolos TCP/IP[27] para las comunicaciones. Fue creada para facilitar la comunicación entre el gobierno y los investigadores. Internet proporciona la infraestructura necesaria para la comunicación e intercambio de información, haciendo posible la existencia de muchos servicios, entre los que se encuentran: El correo electrónico, la transferencia de archivos, acceso a sistemas remotos, conferencias interactivas, grupos de noticias y acceso a la red global internet. Los sistemas operativos juegan un papel muy importante; según lo visto en las siete unidades anteriores, se puede afirmar que el corazón de internet son los sistemas operativos, ya que sin ellos sería imposible poder acceder a todas las ventajas que ofrece a los cientos y aún miles

[27] El nombre TCP/IP proviene de dos de los protocolos más importantes de la familia de protocolos internet, el *Transmission Control Protocol* (TCP) y el internet Protocol (IP). Vid. Raya, José Luis (2001). *TCP/IP para Windows 2000 Server.* Bogotá: Alfaomega, pág. 59.

de computadoras conectadas entre sí. La comunicación de computadoras en internet se realiza mediante la transmisión de paquetes. Cada paquete contiene en su estructura la dirección IP (Protocolo Internet) de origen y destino de la computadora conectada a la red. Cuando el paquete llega al ruteador (equipo de interconexión de redes) éste extrae la dirección de destino utilizada para decidir por cuál ruta de internet debe enviarlo a la computadora destino. Los ruteadores contienen tablas de ruteo en las que se almacena la información sobre las direcciones IP de las computadoras conectadas a la red y se actualizan constantemente.

Servicios de red

Las redes de computadoras ofrecen servicios y procesos a las computadoras y dispositivos que lo requieran; de manera general los protocolos de conectividad de redes y el tráfico de datos que soportan se clasifican en:

1. Servicios orientados a la conexión

Implica el uso de una trayectoria específica que se establece de manera permanente durante el tiempo que dura la conexión. Tiene tres fases: el establecimiento de la conexión, la transferencia de datos y la terminación de la conexión. Los servicios orientados a la conexión son muy útiles para la transmisión de aplicaciones que no toleran retardos y secuenciación de paquetes, los servicios de voz y video se suelen basar en servicios orientados a la conexión.

2. Servicios NO orientados a la conexión

Este servicio se basa en la selección dinámica de la trayectoria y del ancho de banda, logrando con esto que el tráfico sea ruteado y evite su paso por fallas en la red. Este tipo de servicio es muy útil en la transmisión de aplicaciones que pueden tolerar ciertos retardos y resecuenciación de paquetes. Las aplicaciones de datos se basan en servicios no orientados a la conexión (Ford, 1998: 21-23).

Aunque los servicios de Internet más populares son: la navegación por web, correo electrónico, chat, etc., no se limitan a ello, ya que en la realidad se concibe a Internet como un sistema distribuido con alcance mundial. Los servicios más comunes son:

Correo electrónico (e-mail). Es una forma de comunicación que tiene sus propias convenciones y estilos. Utiliza el protocolo SNMP (Simple Mail Transfer Protocol), para el envío y recepción de mensajes.
Voz sobre IP. Transmisión de voz sobre el protocolo IP.
FTP (*File Transfer Protocol*). Protocolo que permite enviar y recibir archivos entre equipos conectados en red.
DNS (*Domain Name Service*). Servicio de nombres de dominio que permite relacionar nombres de computadoras y destinos de servicios, (como el e-mail) con las direcciones electrónicas IP.
WHAIS (*Wide Area Information Service*). Servicio de información basado en bases de datos que permiten su rápida localización.
Finger. Servicio de identificación de usuarios.
WWW. Servicio Web basado en HTTP (Hyper *Text Transfer Protocol*) para navegar en Internet.
Telnet. Protocolo que permite conectar equipos remotos a través de la red emulando una terminal del equipo al que se conecta.
NFS (*Network File System*). Permite conectar equipos que están físicamente separados para compartir directorios y discos por medio de la técnica RPC (*Remote Procedure Call*) como si fueran recursos locales.
NIS (*Network Information Services*). Servicios de información de red que permite que varios sistemas compartan una base de datos con un esquema de seguridad (*password file*) lo que facilita su gestión centralizada.
Servicios R. Permiten ejecutar comandos como; rlogin, rsh, etcétera en sistemas remotos sin requerir *password*.

RESUMEN DE LA UNIDAD

El sistema operativo es más que el conjunto de sus partes, es la cooperación ordenada de todos los elementos de *hardware* y *software*. Cuando se da prioridad a uno de sus elementos es generalmente a expensas de los demás. Por lo que los administradores deberán medir constantemente el rendimiento del sistema y utilizar los mecanismos apropiados para comprobar y mejorar su rendimiento. El módulo del *kernel* que controla a un dispositivo se denomina ***driver*** o manejador de dispositivo y está ubicado entre el *hardware* y la aplicación. La importancia de programar un driver es principalmente para dar soporte a un nuevo *hardware* o *software* y mantener productos propios. Cada dispositivo de entrada/salida, o cada clase de dispositivos, tiene un *driver* asociado en el sistema operativo que incluye: código independiente del dispositivo para proporcionar al nivel superior del sistema operativo una interfaz de alto nivel y el código dependiente del dispositivo necesario para programar el controlador del dispositivo a través de sus registros y datos. Un sistema operativo de red (NOS) es una computadora que realiza funciones de servidor que proporciona servicios a las estaciones de trabajo conectadas en red conocidas como clientes. Para comunicarse utilizan la infraestructura de telecomunicaciones, protocolos y servicios de red. Las funciones del servidor se activan únicamente cuando el sistema requiere utilizar la red, en caso contrario, se mantiene en estado latente y el sistema operativo funciona como sistema independiente. El enfoque general de un NOS es compartir los recursos en vez de ejecutar aplicaciones; es decir, dejan a las estaciones de trabajo la capacidad de compartir los recursos del servidor (aplicaciones, periféricos, módems).Un sistema distribuido es un conjunto de procesadores que no comparten memoria ni reloj de sincronía. Cada procesador tiene su propia memoria local y los procesadores se comunican por medio de la infraestructura de telecomunicaciones (enlaces

dedicados, líneas telefónicas, etcétera). Existen principalmente dos tipos de sistemas distribuidos: LAN (Local Área Network) y WAN (*Wide Area Network*) cada una con sus características propias. Internet es la red de redes de alcance mundial que utiliza la *suite* de protocolos TCP/IP para las comunicaciones. Internet proporciona la infraestructura necesaria para la comunicación e intercambio de información, haciendo posible la existencia de muchos servicios, entre los que se encuentran: El correo electrónico, la transferencia de archivos, acceso a sistemas remotos, conferencias interactivas, grupos de noticias y acceso a la red global Internet.

BIBLIOGRAFÍA

SUGERIDA

Autor	Capítulo	Páginas
Flynn, Ida	11	270-274
Flynn, Ida	10	252-256
Tanenbaum	12	883-889
Carretero, Jesús	7	363-369
Ford, Merilee	1	3-27

REFERENCIAS BIBLIOGRÁFICAS

BIBLIOGRAFÍA SUGERIDA

Carretero, J. (2001). *Sistemas operativos.* Madrid: McGraw Hill.

Dávila, M. (2009). *Linux y software libre*. México: Alfaomega.

Flynn, I. (2001). *Sistemas operativos* (3ª ed.) México: Thomson Learning.

Ford, M. (1998). Tecnologías de interconectividad de redes. México: Prentice-Hal.

Manual de FreeBSD. Proyecto de Documentación de FreeBSD. (2012). El Proyecto FreeBSD. Consultado el 10 de septiembre de 2012 de: http://www.freebsd.org/doc/es/books/handbook/

Meza Badillo, S. (n.d.) *Sistemas operativos multiusuarios*. Fecha de recuperación 05 de diciembre de 2012, en:

http://fcasua.contad.unam.mx/apuntes/interiores/docs/98/2/sis_operativos.pdf

Microsoft Corporation (2001). *Microsoft Windows 2000 Server*. Madrid: McGraw Hill.

Silberschatz, A., Galvin, P. y Gagne, G. (2002). *Sistemas Operativos* (6ª ed.). México: Limusa Wiley.

Tanenbaum, A. (2003). *Sistemas operativos modernos* (2ª ed.). México: Pearson Educación, 976 pp.

William, R. (2004). *Windows Server 2003*. Madrid: Anaya Multimedia.

BIBLIOGRAFÍA BÁSICA

Carretero Pérez, Jesús, (2002). *Práctica de sistemas operativos, de la base al diseño,* México, Mc. GrawHill.

Carretero Pérez, Jesús, (2000). *Sistemas operativos, una visión aplicada*, México, Mc. GrawHill.

Stallings, William, (2003). *Sistemas Operativos*, (4ª. Edición), México, Prentice Hall.

Flynn, Ida M. y McHoes, Ann, (2002). *Sistemas Operativos*, (3ª. Edición) México, Thomson Learning.

López, Ángel, (2002). *Protocolos de Internet, Diseño e implementación en sistemas UNIX*. México, Alfa omegaRama.

Raya, L., A. Martín y V. Rodrigo, (2003) *Sistemas informáticos multiusuario y en red*, México, Alfa omegaRama.

Sarwar, Syed M., (2003). *El libro de UNIX,* México, Addison Wesley.

Tannenbaum, Andrew S. (2003). *Sistemas Operativos Modernos*, (2a. Edición). México: Prentice Hall.

Tiznado, Marco Antonio. (2002). *Sistemas Operativos.* México: Mc. GrawHill.

BIBLIOGRAFÍA COMPLEMENTARIA

Fine, Leonard H., (2002) *Seguridad en centros de cómputo, Políticas y procedimientos*. México, Trillas.

Raya, José Luis y Elena Raya. (2002) *Netware 5. Instalación, configuración y administración*, España: Alfa omegaRama.

Raya, Cabrera José Luis y Cristina Raya Pérez,Netware (2002). *4.11 Intranetware. Instalación, configuración y administración de una red Novell.* España, Alfa omega Rama.

Raya, José Luis. (2002). *La seguridad de una red con Netware 5.* España, Alfa omega Rama.

Raya, José Luis (2002). *Redes locales y TCP/IP.* España, Alfa omega Rama.

Plan 2012

2016

actualizado