

UNIVERSIDAD POLITÉCNICA DE VALENCIA  
DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN



---

# **Administración de Sistemas**

**RedHat Linux 7.2  
y Windows 2000**

---

**Agustín Espinosa Minguet  
Andrés M. Terrasa Barrena  
Fernando Ferrer García  
Álvaro Álvarez Rodríguez**

Curso Académico 2003/2004  
Valencia, 22 de Septiembre de 2003



# Índice general

---

<b>1. Introducción</b>	<b>1</b>
1.1. Profesorado . . . . .	3
1.2. Programa . . . . .	3
1.3. Evaluación . . . . .	3
1.4. Horarios . . . . .	4
1.5. Documentación y Páginas Web . . . . .	4
<b>2. El Sistema de Nombres de Dominio</b>	<b>5</b>
2.1. Funcionamiento de DNS . . . . .	7
2.1.1. El Espacio de Nombres de Dominio . . . . .	7
2.1.2. El Espacio de Nombres de Dominio de Internet . . . . .	8
2.1.3. Delegación . . . . .	8
2.1.4. Servidores de Nombres y Zonas . . . . .	9
2.1.5. Resolución de Nombres . . . . .	9
2.2. Configuración de DNS . . . . .	10
2.2.1. Registros de Recursos (RR) . . . . .	11
Registro de Recurso SOA . . . . .	11
Registro de Recurso NS . . . . .	13
Registro de Recurso A . . . . .	13
Registro de Recurso PTR . . . . .	13
Registro de Recurso CNAME . . . . .	14

Registro de Recurso MX . . . . .	14
Registro de Recurso SRV . . . . .	14
2.2.2. Definición de la Delegación . . . . .	15
2.2.3. Tipos de Zona . . . . .	16
Zona de Búsqueda Directa . . . . .	16
Zona de Búsqueda Inversa . . . . .	16
Sugerencias de los Servidores del Dominio Raíz . . . . .	17
2.2.4. Transferencias de Zona . . . . .	17
Transferencia Completa de Zona . . . . .	17
Transferencia Incremental de Zona . . . . .	18
Notificación DNS . . . . .	18
2.2.5. Actualizaciones Dinámicas . . . . .	18
<b>3. Niveles de ejecución en Linux</b>	<b>19</b>
3.1. Inicio y Detención de un Sistema Linux . . . . .	21
3.2. El Concepto de Nivel de Ejecución . . . . .	21
3.3. Ficheros y Directorios de Configuración de los Niveles de Ejecución . . . . .	22
3.4. Secuencia de Arranque de un Sistema Linux . . . . .	23
3.5. Secuencia de Entrada en un Nivel de Ejecución . . . . .	23
3.6. Servicios independientes de los Niveles de Ejecución . . . . .	24
3.7. Configuración de los Niveles de Ejecución . . . . .	25
<b>4. Usuarios y Protección en Linux</b>	<b>27</b>
4.1. La Tabla de Usuarios . . . . .	29
4.2. La Extensión de la Tabla de Usuarios . . . . .	29
4.3. La Tabla de Grupos . . . . .	30
4.4. Procedimientos para la Creación de Grupos y Usuarios . . . . .	31
4.5. Atributos de Protección de los Procesos . . . . .	31
4.6. Atributos de Protección de los Ficheros . . . . .	32
4.7. Las Reglas de Protección Básicas . . . . .	33
4.8. Cambio de Atributos de Protección en Ficheros . . . . .	34
4.9. Los Bits SETUID y SETGID en Ficheros Ejecutables . . . . .	35

4.10. El Bit SETGID en Directorios . . . . .	35
4.11. La Máscara de Creación de Ficheros . . . . .	35
4.12. La Estrategia de los Grupos Privados . . . . .	36
4.13. Sintaxis y Funcionamiento de los Mandatos Unix Relacionados . . . . .	36
<b>5. Protección Local en Windows 2000</b>	<b>41</b>
5.1. Conceptos de Usuario y de Cuenta de Usuario . . . . .	43
5.2. Grupos de Usuarios . . . . .	44
5.3. El Modelo de Protección de Windows 2000 . . . . .	45
5.4. Atributos de Protección de los Procesos . . . . .	46
5.5. Derechos de Usuario . . . . .	47
5.5.1. Otras Directivas de Seguridad . . . . .	48
5.6. Atributos de Protección de los Recursos . . . . .	48
5.6.1. Permisos Estándar y Permisos Individuales . . . . .	50
5.7. Reglas de Protección . . . . .	51
<b>6. Montaje de Dispositivos en Linux</b>	<b>55</b>
6.1. Introducción . . . . .	57
6.2. Utilización Básica . . . . .	57
6.3. La tabla de Montaje de Dispositivos . . . . .	58
6.4. Otros mandatos relacionados . . . . .	60
<b>7. Dominios en Linux</b>	<b>61</b>
7.1. Concepto de Dominio . . . . .	61
7.2. Dominios en Linux con OpenLDAP . . . . .	62
7.2.1. Introducción . . . . .	62
7.2.2. Servicios de Directorio y LDAP . . . . .	62
7.2.3. Configuración Básica de OpenLDAP . . . . .	65
Configuración de OpenLDAP con Servidor Unico . . . . .	66
Configuración de OpenLDAP con Múltiples Servidores . . . . .	72
7.2.4. Implementación de un Dominio Linux con OpenLDAP . . . . .	73
7.2.5. Herramientas Gráficas de Administración . . . . .	75

7.3. Network File System (NFS) . . . . .	76
7.3.1. Cómo Funciona NFS . . . . .	77
7.3.2. Instalación y Configuración del Cliente NFS . . . . .	77
7.3.3. Instalación y Configuración del Servidor NFS . . . . .	78
<b>8. Dominios en Windows 2000</b>	<b>81</b>
8.1. Concepto de Dominio en Windows 2000 . . . . .	83
8.2. Implementación de Dominios: el Directorio Activo . . . . .	85
8.2.1. Concepto de Directorio . . . . .	85
8.2.2. Concepto de Dominio Según el Directorio Activo . . . . .	86
8.2.3. Múltiples Dominios en la Misma Organización . . . . .	86
8.3. Principales Objetos que Administra un Dominio . . . . .	87
8.3.1. Usuarios Globales . . . . .	87
8.3.2. Grupos Globales . . . . .	88
8.3.3. Equipos . . . . .	89
8.3.4. Unidades Organizativas . . . . .	90
8.4. Compartición de Recursos entre Sistemas 2000 . . . . .	91
8.5. Confianzas entre Dominios Windows 2000 . . . . .	92
8.6. Mandatos Windows 2000 para Compartir Recursos . . . . .	93
8.7. Delegación de la Administración a Unidades Organizativas . . . . .	94
<b>9. Configuración de Samba</b>	<b>97</b>
9.1. ¿Qué es samba? . . . . .	99
9.2. Configuración de Samba . . . . .	100
9.3. Niveles de Seguridad . . . . .	101
9.4. Configuración de Samba con el Nivel de Seguridad domain . . . . .	103
9.5. Tratamiento de los Accesos como Invitado . . . . .	103
9.6. El Sistema de Ficheros SMB para Linux . . . . .	104
9.7. Opciones del Servidor Samba . . . . .	105
9.8. Opciones del Recurso . . . . .	105

<b>A. Ejemplo de Configuración DNS</b>	<b>107</b>
A.1. Fichero /etc/named.conf . . . . .	109
A.2. Fichero /var/named/db.punto . . . . .	110
A.3. Fichero /var/named/db.in-addr.arpa . . . . .	111
A.4. Fichero /var/named/db.admon.com . . . . .	112
A.5. Fichero /var/named/db.42.158 . . . . .	113
A.6. Fichero /var/named/named.local . . . . .	115





# Índice de figuras

---

3.1. Editor de niveles de ejecución de RedHat Linux. . . . .	26
7.1. Estructura de directorio del dominio <code>admon.com</code> . . . . .	64
7.2. Vista de la herramienta Directory Administrator. . . . .	75
9.1. Ejemplo de fichero <code>/etc/samba/smb.conf</code> . . . . .	100



# Índice de cuadros

---

3.1. Niveles de ejecución en RedHat Linux. . . . .	21
4.1. Significado de los bits de permiso en Unix. . . . .	33
5.1. Derechos más importantes en Windows 2000 . . . . .	47
5.2. Permisos estándar sobre carpetas y archivos en Windows 2000 . . . . .	51
5.3. Permisos individuales en Windows 2000 . . . . .	52
5.4. Correspondencia entre permisos estándar e individuales en Windows 2000. . . . .	53
6.1. Principales tipos de particiones utilizables desde Linux. . . . .	58
6.2. Interpretación del fichero <code>/etc/fstab</code> . . . . .	59
6.3. Opciones más comunes en el montaje de dispositivos en Linux. . . . .	59
7.1. Opciones más usuales en la exportación de directorios mediante NFS. . . . .	79
9.1. Secciones globales en el fichero <code>/etc/smb.conf</code> . . . . .	101
9.2. Distintos comportamientos del acceso a un servidor Samba. . . . .	104
9.3. Principales opciones de la sección <code>[global]</code> de Samba. . . . .	105
9.4. Principales opciones de recurso de Samba. . . . .	106



# Capítulo 1

## Introducción

### Índice General

---

1.1. Profesorado . . . . .	3
1.2. Programa . . . . .	3
1.3. Evaluación . . . . .	3
1.4. Horarios . . . . .	4
1.5. Documentación y Páginas Web . . . . .	4

---



## 1.1. Profesorado

- Agustín Espinosa Minguet (aespinos@dsic.upv.es)
- Andrés Terrasa Barrena (aterrasa@dsic.upv.es)
- Alvaro Alvarez Rodríguez (aarodri@dsic.upv.es)

## 1.2. Programa

- 1) Instalación de RedHat Linux.
- 2) Instalación de Windows 2000.
- 3) Servicio de Nombres de Dominio en Linux.
- 4) Servicio de Nombres de Dominio en Windows 2000.
- 5) Seguridad y Protección en Linux.
- 6) Seguridad y Protección en Windows 2000.
- 7) Dominios y Sistemas de Archivos en Red en Linux.
- 8) Dominios y Sistemas de Archivos en Red en Windows 2000.
- 9) Interoperatividad entre Linux y Windows 2000 (Samba).
- 10) Políticas de Grupo en Windows 2000.

## 1.3. Evaluación

### a) Convocatoria Ordinaria

- Actividad de laboratorio 50 %.
  - Cada no asistencia a prácticas resta un 10 % de la nota de prácticas.
  - Cada práctica es evaluada con calificación al grupo de Suspenso, Notable o Sobresaliente.
  - Deben aprobarse las prácticas para aprobar la asignatura.
  - Prácticas recuperables en otro turno que realice la práctica a recuperar.
- Examen escrito 50 %
  - Individual.
  - Debe aprobarse el examen escrito para aprobar la asignatura.

### b) Convocatoria Extraordinaria

- Examen escrito (50 %) para alumnos con prácticas aprobadas en Junio.
- Examen escrito (100 %) con ejercicio práctico para el resto.

## 1.4. Horarios

TEORÍA			
Día	Hora	Aula	Profesor
Martes	De 15 a 17	*****	Agustín Espinosa
Miércoles	De 12:30 a 14:30	*****	Agustín Espinosa
LABORATORIO			
Martes	De 8:30 a 10:20	Laboratorio 4 DSIC	Álvaro Álvarez
Martes	De 10:30 a 12:30	Laboratorio 4 DSIC	Agustín Espinosa
Miércoles	De 15:00 a 17:00	Laboratorio 4 DSIC	Andrés Terrasa
Miércoles	De 17:00 a 19:00	Laboratorio 4 DSIC	Álvaro Álvarez
Viernes	De 10:30 a 12:30	Laboratorio 4 DSIC	Agustín Espinosa
Viernes	De 12:30 a 14:30	Laboratorio 4 DSIC	Agustín Espinosa

## 1.5. Documentación y Páginas Web

La documentación de las asignaturas consiste en este documento, que incluye los temas teóricos, más un documento semanal que describe la actividad práctica a realizar en cada sesión de laboratorio.

Toda la documentación podrá encontrarse en las páginas web de las asignaturas, así como comprarse en reprografía.

Las páginas web de las asignaturas son sus *microwebs UPV*. Se requiere acreditación como alumno matriculado para poder acceder a la documentación.



# Capítulo 2

## El Sistema de Nombres de Dominio

### Índice General

---

<b>2.1. Funcionamiento de DNS</b>	<b>7</b>
2.1.1. El Espacio de Nombres de Dominio	7
2.1.2. El Espacio de Nombres de Dominio de Internet	8
2.1.3. Delegación	8
2.1.4. Servidores de Nombres y Zonas	9
2.1.5. Resolución de Nombres	9
<b>2.2. Configuración de DNS</b>	<b>10</b>
2.2.1. Registros de Recursos (RR)	11
2.2.2. Definición de la Delegación	15
2.2.3. Tipos de Zona	16
2.2.4. Transferencias de Zona	17
2.2.5. Actualizaciones Dinámicas	18

---



## 2.1. Funcionamiento de DNS

El Domain Name System (DNS) o Sistema de Nombres de Dominio permite a los usuarios de una red TCP/IP utilizar nombres jerárquicos y descriptivos para localizar fácilmente ordenadores (hosts) y otros recursos en dicha red, evitando de esta manera tener que recordar la dirección IP de cada ordenador al que se desea acceder. En esencia, DNS es una base de datos distribuida que contiene principalmente asociaciones de nombres simbólicos (de hosts) a direcciones IP. El hecho de que sea distribuida permite delegar el control sobre diferentes segmentos de la base de datos a distintas organizaciones, pero siempre de forma que los datos de cada segmento están disponibles en toda la red, a través de un esquema cliente-servidor.

Los programas denominados servidores de nombres (name servers) constituyen la parte servidora del esquema cliente-servidor. Los servidores de nombres contienen información sobre algunos segmentos de la base de datos y los ponen a disposición de los clientes, llamados solucionadores o resolvers.

### 2.1.1. El Espacio de Nombres de Dominio

La base de datos distribuida de DNS está indexada por nombres de dominio. Cada nombre de dominio es esencialmente una trayectoria en un árbol invertido denominado *espacio de nombres de dominio*. La estructura jerárquica del árbol es similar a la estructura del sistema de ficheros UNIX. El árbol tiene una única raíz en el nivel superior llamada raíz (root). Cada nodo del árbol puede ramificarse en cualquier número de nodos de nivel inferior. La profundidad del árbol está limitada a 127 niveles.

Cada nodo en el árbol se identifica mediante una etiqueta no nula que puede contener hasta 63 caracteres, excepto el nodo raíz, identificado mediante una etiqueta nula. El nombre de dominio completo de cualquier nodo está formado por la secuencia de etiquetas que forman la trayectoria desde dicho nodo hasta la raíz, separando cada etiqueta de la siguiente mediante un punto. De esta forma, el nombre del nodo especifica de forma unívoca su localización en la jerarquía. A este nombre de dominio completo o absoluto se le conoce como *nombre de dominio completamente cualificado* o Fully Qualified Domain Name (FQDN). Al ser nula la etiqueta que identifica el nodo raíz, el FQDN de cualquier nodo del árbol siempre acaba con un punto. La única restricción que se impone en el árbol de nombres es que los nodos hijos del mismo padre tengan etiquetas diferentes.

En el esquema jerárquico de nombres DNS, se denomina *dominio* a cualquier subárbol del espacio de nombres de dominio. De esta forma, cada dominio puede contener, a su vez, otros dominios. Generalmente, los hosts están representados por las hojas del árbol, aunque es posible nombrar a un host con una etiqueta correspondiente a un nodo intermedio del árbol (en este caso, tendríamos un dominio y un nodo que se llaman igual).

La información sobre los nombres de dominio DNS se guarda mediante los denominados *registros de recursos* en los servidores DNS de la red. Concretamente, cada

servidor DNS contiene los registros de recursos necesarios para responder a las consultas sobre la parte del espacio de nombres en la que tiene autoridad.

### 2.1.2. El Espacio de Nombres de Dominio de Internet

El estándar DNS no impone muchas reglas sobre las etiquetas de los nombres de dominio, ni tampoco asocia un significado determinado a las etiquetas de un determinado nivel del espacio de nombres. Cuando manejamos una parte de este espacio, podemos decidir el significado y la sintaxis de nuestros nombres de dominio. Sin embargo, en el espacio de nombres Internet existente, se ha impuesto una estructura de nombres bien definida, especialmente en los dominios de primer nivel.

Los dominios originales de primer nivel dividían originalmente el espacio de nombres de Internet en siete dominios: com, edu, gov, mil, net, org, e int. Posteriormente, para acomodar el crecimiento y la internacionalización de Internet, se reservaron nuevos dominios de primer nivel que hacían referencia a países individuales.

Actualmente, los dominios originales se denominan *dominios de primer nivel genéricos* y han surgido nuevos nombres que se ajustan a los tiempos que corren.

### 2.1.3. Delegación

Es importante resaltar que el objetivo principal del diseño del sistema de nombres de dominio fue su administración descentralizada. Este objetivo se consigue a través de la *delegación*. La delegación de dominios funciona de forma parecida a la delegación de tareas en una organización. Un responsable de proyecto divide el proyecto en pequeñas tareas y asigna (delega) la responsabilidad de las mismas a diferentes empleados.

De la misma forma, una organización que administra un dominio puede dividirla en subdominios. Cada subdominio puede ser delegado a diferentes organizaciones, lo cual implica que esa organización será responsable de mantener los datos (registros de recursos) de ese subdominio. Esa organización puede libremente cambiar los datos e incluso volver a dividir el dominio delegado en subdominios y delegarlos. El dominio padre solamente contiene enlaces a los responsables del subdominio delegado, de forma que pueda hacer referencia a ellos cuando se le planteen consultas sobre nombres en dicho subdominio delegado.

Realmente, la subdivisión de un dominio en subdominios y la delegación de dichos subdominios son cosas distintas. En primer lugar, un dominio que tenga capacidad de autogestión (autoridad), siempre puede decidir subdividirse en diferentes subdominios, manteniendo él en principio la autoridad sobre todos ellos. Posteriormente, la organización que gestiona el dominio puede decidir además delegar la autoridad de algunos (o todos) sus subdominios en otras organizaciones. La delegación es una acción que siempre decide el dominio padre, y éste puede revocarla cuando desee, volviendo a retomar la autoridad sobre el subdominio que había delegado.

#### 2.1.4. Servidores de Nombres y Zonas

Como se ha dicho anteriormente, los programas que almacenan información sobre el espacio de nombres de dominio se denominan servidores de nombres. En virtud de la delegación mencionada anteriormente, cada servidor de nombres posee generalmente información completa sobre una *parte contigua* del espacio de nombres (generalmente un dominio, potencialmente dividido en subdominios). Dicha parte del espacio se denomina *zona*, y se dice que el servidor de nombres tiene *autoridad* sobre ella. En realidad, un mismo servidor de nombres puede tener autoridad sobre múltiples zonas, y obtiene la información que describe la zona (los registros de recursos) o bien de un fichero local o bien de otro servidor de nombres.

Entender la diferencia entre una zona y un dominio es importante. Todos los dominios de primer nivel, y la mayoría de dominios de segundo nivel, se dividen en unidades más pequeñas y manejables gracias a la delegación. Estas unidades se denominan zonas y contienen una serie de registros almacenados en un servidor. Sin embargo, las zonas no son dominios. Un dominio es un subárbol del espacio de nombres, mientras que una zona es una parte del espacio de nombres DNS que se almacena generalmente en un fichero y que puede contener información sobre múltiples dominios.

DNS define dos tipos de servidores de nombres que mantienen información sobre el espacio de nombres: primarios (maestros) y secundarios (esclavos). Un servidor de nombres primario para una zona lee los datos de la zona desde un fichero que él mantiene. Un servidor de nombres secundario para una zona obtiene los datos de la zona desde otro servidor de nombres que es autoritario para la zona, llamado servidor maestro. Normalmente el servidor maestro es el servidor primario de la zona, pero esto no es un requisito ya que un servidor secundario puede cargar los datos desde otro secundario.

Cuando un servidor de nombres secundario se inicia, éste se pone en contacto con su servidor maestro y, si es necesario, inicia una transferencia de zona, es decir, una actualización de su información sobre la zona (ver Sección 2.2.4). Además, periódicamente el servidor secundario contacta con el servidor maestro para ver si los datos de zona han cambiado. Tanto el servidor primario como el secundario poseen autoridad sobre la zona. Definir servidores secundarios proporciona tolerancia a errores y reduce la carga en el servidor primario de la zona.

#### 2.1.5. Resolución de Nombres

Los clientes DNS utilizan bibliotecas llamadas solucionadores (resolvers) que efectúan las consultas DNS a los servidores en nombre del cliente.

Los servidores de nombres son los expertos en obtener información del espacio de nombres de dominio. Es decir, no solamente responden los datos referentes a las zonas sobre los que tienen autoridad, sino que pueden también buscar información a través del espacio de nombres de dominio para encontrar datos sobre los que no son autoritarios. A este proceso se le denomina *resolución de nombres*. Por ese motivo,

existen servidores de nombres que no mantienen información sobre ninguna zona, y únicamente sirven para responder consultas de los clientes (resolvers) sobre cualquier dominio. Este tipo de servidores DNS se denomina *cache only*.

Ya que el espacio de nombres está estructurado como un árbol invertido, un servidor de nombres necesita únicamente los nombres de dominio y las direcciones de los servidores de nombres raíz para encontrar cualquier punto en el árbol. Los servidores raíz conocen dónde se encuentran los servidores de nombres con autoridad para los dominios de primer nivel.

Cuando se solicita una consulta a cualquier nombre de dominio, los servidores raíz pueden al menos proporcionar los nombres y direcciones de los servidores de nombres autoritarios para el dominio de primer nivel al que pertenece el nombre de dominio buscado. Y los servidores de nombres de primer nivel pueden proporcionar la lista de servidores de nombres autoritarios para el dominio de segundo nivel al que pertenece el nombre de dominio buscado. De esta forma, cada servidor de nombres consultado va proporcionando la información más próxima a la respuesta buscada, o proporciona la propia respuesta.

Como conclusión hay que resaltar la importancia que tienen los servidores de nombres raíz en el proceso de resolución. Por esta razón, el sistema de nombres de dominio proporciona mecanismos de caché para ayudar a reducir la carga que supondría el proceso de resolución sobre los servidores raíz. Si todos los servidores raíz de Internet fallaran por un largo período de tiempo, toda la resolución en Internet fallaría. Para protegerse, Internet posee 13 servidores de nombres raíz repartidos por diferentes partes de la Red.

## 2.2. Configuración de DNS

Los estándares de DNS no especifican la estructura de datos interna en que deben almacenarse los registros de recursos (registros de la base de datos DNS), y por tanto existen varias implementaciones que son diferentes en este sentido. Por regla general, los servidores guardan la información sobre las zonas en ficheros en texto plano sin formato. Los nombres de los archivos son arbitrarios y se especifican en la configuración del servidor DNS.

Por ejemplo, en la implementación habitual de DNS en el mundo UNIX, BIND (Berkeley Internet Name Domain), se utiliza los nombres de archivo siguientes para almacenar los registros de cada zona:

- `Db.dominio` (zona de resolución directa)
- `Db.direccion` (zona de resolución inversa)
- `Db.cache` (sugerencias de servidores raíz)
- `Db.127.0.0.1` (resolución inversa de bucle cerrado)

Por el contrario, la configuración predeterminada del servidor DNS de Microsoft Windows 2000 no utiliza los mismos nombres de archivo que BIND, sino que usa la nomenclatura `nombre_zona.dns`. Por otra parte, en Windows 2000, la base de datos DNS puede integrarse con la base de datos de Active Directory, en cuyo caso dicha información participa de los mismos mecanismos de almacenamiento y replicación que el resto de información contenida en el Directorio Activo.

### 2.2.1. Registros de Recursos (RR)

Para resolver nombres, los servidores consultan sus zonas. Las zonas contienen *registros de recursos* que constituyen la información de recursos asociada al dominio DNS. Por ejemplo, ciertos registros de recursos asignan nombres descriptivos a direcciones IP.

El formato de cada registro de recursos es el siguiente:

Propietario	TTL	Clase	Tipo	RDATA
-------------	-----	-------	------	-------

donde:

- **Propietario:** nombre de host o del dominio DNS al que pertenece este recurso. Puede contener un nombre de host/dominio (completamente cualificado o no), el símbolo “@” (que representa el nombre de la zona que se está describiendo) o una cadena vacía (en cuyo caso equivale al propietario del registro de recursos anterior).
- **TTL:** (Time To Live) Tiempo de vida, generalmente expresado en segundos, que un servidor DNS o un resolver debe guardar en caché esta entrada antes de descartarla. Este campo es opcional. También se puede expresar mediante letras indicando días (d), horas (h), minutos (m) y segundos (s). Por ejemplo: “2h30m”.
- **Clase:** define la familia de protocolos en uso. Suele ser siempre “IN”, que representa Internet.
- **Tipo:** identifica el tipo de registro.
- **RDATA:** los datos del registro de recursos.

A continuación se describen los principales tipos de registros de recursos: SOA, NS, A, PTR, CNAME, MX y SRV.

#### Registro de Recurso SOA

Cada zona contiene un registro de recursos denominado Inicio de Autoridad o SOA (Start Of Authority) al comienzo de la zona. Los registros SOA incluyen los siguientes campos (sólo se incluyen los que poseen un significado específico para el tipo de registro):

- Propietario: nombre de dominio de la zona.
- Tipo: “SOA”.
- Servidor primario (maestro) de la zona.
- Persona responsable: contiene la dirección de correo electrónico del responsable de la zona. En esta dirección de correo, se utiliza un punto en el lugar del símbolo “@”.
- Número de serie: muestra el número de versión de la zona, es decir, un número que sirve de referencia a los servidores secundarios de la zona para saber cuándo deben proceder a una actualización de su base de datos de la zona (o *transferencia de zona*). Cuando el número de serie del servidor secundario sea *menor* que el número del maestro, esto significa que el maestro ha cambiado la zona, y por tanto el secundario debe solicitar al maestro una transferencia de zona. Por tanto, este número debe ser incrementado (manualmente) por el administrador de la zona cada vez que realiza un cambio en algún registro de la zona (en el servidor maestro).
- Actualización: muestra cada cuánto tiempo un servidor secundario debe ponerse en contacto con el maestro para comprobar si ha habido cambios en la zona.
- Reintentos: define el tiempo que el servidor secundario, después de enviar una solicitud de transferencia de zona, espera para obtener una respuesta del servidor maestro antes de volverlo a intentar.
- Caducidad: define el tiempo que el servidor secundario de la zona, después de la transferencia de zona anterior, responderá a las consultas de la zona antes de descartar la suya propia como no válida.
- TTL mínimo: este campo especifica el tiempo de validez (o de vida) de las respuestas “negativas” que realiza el servidor. Una respuesta negativa significa que el servidor contesta que un registro no existe en la zona.

Hasta la versión 8.2 de BIND, este campo establecía el tiempo de vida por defecto de todos los registros de la zona que no tuvieran un campo TTL específico. A partir de esta versión, esto último se consigue con una *directiva* que debe situarse al principio del fichero de la zona. Esta directiva se especifica así:

\$TTL tiempo

Por ejemplo, un tiempo de vida por defecto de 30 minutos se establecería así:

\$TTL 30m



Un ejemplo de registro SOA sería el siguiente:

```
admon.com.  IN  pc0100.admon.com  hostmaster.admon.com.
(
    1      ; número de serie
    3600   ; actualización 1 hora
    600    ; reintentar 10 minutos
    86400  ; caducar 1 día
    60     ; TTL 1 minuto
)
```

### Registro de Recurso NS

El registro de recursos NS (Name Server) indica los servidores de nombres autorizados para la zona. Cada zona debe contener registros indicando tanto los servidores principales como los secundarios. Por tanto, cada zona debe contener, como mínimo, un registro NS.

Por otra parte, estos registros también se utilizan para indicar quiénes son los servidores de nombres con autoridad en subdominios delegados, por lo que la zona contendrá al menos un registro NS por cada subdominio que haya delegado.

Ejemplos de registros NS serían los siguientes:

```
admon.com.          IN  NS      pc0100.admon.com.
valencia.admon.com. IN  NS      pc0102.valencia.admon.com.
```

### Registro de Recurso A

El tipo de registro de recursos A (Address) asigna un nombre de dominio completamente cualificado (FQDN) a una dirección IP, para que los clientes puedan solicitar la dirección IP de un nombre de host dado.

Un ejemplo de registro A que asignaría la dirección IP 158.42.178.1 al nombre de dominio `pc0101.valencia.admon.com.`, sería el siguiente:

```
pc0101.valencia.admon.com.  IN  A      158.42.178.1
```

### Registro de Recurso PTR

El registro de recursos PTR (PoinTeR) o puntero, realiza la acción contraria al registro de tipo A, es decir, asigna un nombre de dominio completamente cualificado a una dirección IP. Este tipo de recursos se utilizan en la denominada *resolución inversa*, descrita en la Sección 2.2.3.

Un ejemplo de registro PTR que asignaría el nombre `pc0101.valencia.admon.com.` a la dirección IP `158.42.178.1` sería el siguiente:

```
1.178.42.158.in-addr.arpa.  IN  PTR  pc0101.admon.valencia.com.
```

## Registro de Recurso CNAME

El registro de nombre canónico (CNAME, Canonical NAME) crea un alias (un sinónimo) para el nombre de dominio especificado.

Un ejemplo de registro CNAME que asignaría el alias `controlador` al nombre de dominio `pc0102.valencia.admon.com`, sería el siguiente:

```
controlador.valencia.admon.com.
                               IN      CNAME    pc0101.valencia.admon.com.
```

## Registro de Recurso MX

El registro de recurso de intercambio de correo (MX, Mail eXchange) especifica un servidor de intercambio de correo para un nombre de dominio. Puesto que un mismo dominio puede contener diferentes servidores de correo, el registro MX puede indicar un valor numérico que permite especificar el orden en que los clientes deben intentar contactar con dichos servidores de correo.

Un ejemplo de registro de recurso MX que define al servidor `pc0100` como el servidor de correo del dominio `admon.com`, sería el siguiente:

```
admon.com.  IN      MX      0      pc0100.admon.com.
```

## Registro de Recurso SRV

Con registros MX se puede especificar varios servidores de correo en un dominio DNS. De esta forma, cuando un proveedor de servicio de envío de correo necesite enviar correo electrónico a un host en el dominio, podrá encontrar la ubicación de un servidor de intercambio de correo. Sin embargo, esta no es la forma de resolver los servidores que proporcionan otros servicios de red como WWW o FTP.

Los registros de recurso de servicio (SRV, SeRVice) permiten especificar de forma genérica la ubicación de los servidores para un servicio, protocolo y dominio DNS determinados.

El formato de un registro SRV es el siguiente:

```
servicio.protocolo.nombre  TTL  clase  SRV
                             prioridad  peso  puerto  destino
```

donde:

- El campo `servicio` especifica el nombre de servicio: `http`, `telnet`, etc.
- El campo `protocolo` especifica el protocolo utilizado: `TCP` o `UDP`.
- `nombre` define el nombre de dominio al que hace referencia el registro de recurso SRV.
- Los campos `TTL` y `clase` ha sido definidos anteriormente.
- `prioridad`: los clientes intentarán ponerse en contacto primero con el host que tenga el valor de prioridad más bajo, luego con el siguiente y así sucesivamente.
- `peso`: es un mecanismo de equilibrio de carga.
- `puerto`: muestra el puerto del servicio en el host.
- `destino`: muestra el nombre de dominio completo para la máquina compatible con ese servicio.

Un ejemplo de registros SRV para los servidores Web del dominio `admon.com.`, sería:

```
http.tcp.admon.com.  IN  SRV  0  0  80  www1.admon.com.
http.tcp.admon.com.  IN  SRV  10 0  80  www2.admon.com.
```

### 2.2.2. Definición de la Delegación

Para que una zona especifique que uno de sus subdominios está delegado en una zona diferente, es necesario agregar un *registro de delegación* y, generalmente, el denominado “registro de pegado” (glue record). El registro de delegación es un registro NS en la zona principal (padre) que define el servidor de nombres autorizado para la zona delegada. El registro de pegado es un registro tipo A para el servidor de nombres autorizado para la zona delegada, y es necesario cuando el servidor de nombres autorizado para la zona delegada también es un miembro de ese dominio (delegado).

Por ejemplo, si la zona `admon.com` deseara delegar la autoridad a su subdominio `valencia.admon.com`, se deberían agregar los siguientes registros al archivo de configuración correspondiente de la zona `admon.com`:

```
valencia.admon.com.      IN  NS  pc0102.valencia.admon.com.
pc0102.valencia.admon.com. IN  A   158.42.178.2
```

### 2.2.3. Tipos de Zona

Aunque distintas implementaciones de DNS difieren en cómo configurar las zonas, generalmente existe un fichero que indica sobre qué zonas tiene autoridad el servidor, indicando para cada una el fichero que contiene la información de dicha zona (si el servidor es primario para la zona), o la dirección del servidor maestro a quien preguntar por ella (si es secundario).

En general, existen tres tipos distintos de zonas: zonas de búsqueda directa, zonas de búsqueda inversa y zonas de “sugerencia raíz”. Un servidor DNS puede tener autoridad sobre varias zonas directas e inversas, y necesita poseer información sobre las “sugerencias raíz” si desea responder a sus clientes sobre registros de zonas sobre las que no posee autoridad. A continuación se describe cada tipo brevemente.

#### Zona de Búsqueda Directa

Las zonas de búsqueda directa contienen la información necesaria para resolver nombres en el dominio DNS. Deben incluir, al menos, registros SOA y NS, y pueden incluir cualquier otro tipo de registros de recurso, excepto el registro de recursos PTR.

#### Zona de Búsqueda Inversa

Las zonas de búsqueda inversa contienen información necesaria para realizar las búsquedas inversas. La mayor parte de las consultas proporcionan un nombre y solicitan la dirección IP que corresponde a ese nombre. Este tipo de consulta es el descrito en la zona de resolución directa.

Pero existen ocasiones en que un cliente ya tiene la dirección IP de un equipo y desea determinar el nombre DNS de ese equipo. Esto es importante para los programas que implementan la seguridad basándose en el FQDN que se conecta y también se utiliza para la solución de problemas de red TCP/IP.

Si el único medio de resolver una búsqueda inversa es realizar una búsqueda detallada de todos los dominios en el espacio de nombres DNS, la búsqueda de consulta inversa sería demasiado exhaustiva como para realizarla de forma práctica.

Para solucionar este problema se creó un dominio DNS especial para realizar búsquedas “inversas”, denominado `in-addr.arpa.`. Este dominio utiliza un orden inverso de números en la notación decimal de las direcciones IP. Con esta disposición se puede delegar la autoridad de miembros inferiores del dominio `in-addr.arpa.` a las distintas organizaciones, a medida que se les asigna identificadores de red de clase A, B o C.

### Sugerencias de los Servidores del Dominio Raíz

El archivo de “sugerencias raíz” (root hint), denominado también archivo de sugerencias de caché, contiene la información de host necesaria para resolver nombres fuera de los dominios en los que el servidor posee autoridad. En concreto, este archivo contiene los nombres y las direcciones IP de los servidores DNS del dominio punto (.) o raíz.

#### 2.2.4. Transferencias de Zona

En aquellas zonas en las que existen diferentes servidores de nombres con autoridad (uno principal o maestro y uno o varios secundarios o esclavos), cada vez que se realizan cambios en la zona del servidor maestro, estos cambios deben replicarse a todos los servidores secundarios de esa zona. Esta acción se lleva a cabo mediante un mecanismo denominado transferencia de zona. Existen dos tipos de transferencia de zonas: completa e incremental.

##### Transferencia Completa de Zona

En una transferencia completa de zona, el servidor maestro para una zona transmite toda la base de datos de zona al servidor secundario para esa zona.

Los servidores secundarios siguen los siguientes pasos a la hora de realizar una transferencia de zona:

- El servidor secundario para la zona espera el tiempo especificado en el campo Actualizar del registro SOA y luego le pregunta al servidor maestro por su registro SOA.
- El servidor maestro responde con su registro SOA.
- El servidor secundario para la zona compara el número de serie devuelto con su propio número y si este es mayor que el suyo, solicita una transferencia de zona completa.
- El servidor maestro envía la base de datos de la zona completa al servidor secundario.

Si el servidor maestro no responde, el servidor secundario lo seguirá intentando después del intervalo especificado en el campo Reintentos del registro SOA. Si todavía no hay respuesta después del intervalo que se especifica en el campo Caduca desde la última transferencia de zona, este descarta su zona.

### Transferencia Incremental de Zona

Las transferencias completas de zona pueden consumir gran ancho de banda de la red. Para poder solucionar este problema se define la transferencia incremental de zona, en la cual sólo debe transferirse la parte modificada de una zona.

La transferencia incremental de zona funciona de forma muy similar a la transferencia completa. En este caso, el servidor secundario para la zona comprueba el número de serie del registro SOA del maestro con el suyo, para determinar si debe iniciar una transferencia de zona, la cual en este caso sería incremental (sólo de los cambios realizados).

### Notificación DNS

Con este proceso se pretende que el servidor maestro para la zona notifique los cambios a ciertos servidores secundarios y de esta manera los secundarios podrán comprobar si necesitan iniciar una transferencia de zona. De esta forma se mejora la coherencia de los datos mantenida por todos los servidores secundarios.

#### 2.2.5. Actualizaciones Dinámicas

Originalmente, DNS se diseñó para que solamente admitiera cambios estáticos. De esta forma, sólo el administrador del sistema DNS podía agregar, quitar o modificar los registros de recursos, realizando cambios manuales sobre los ficheros de configuración correspondientes.

El sistema de actualizaciones dinámicas, permite que el servidor principal para la zona pueda configurarse de forma que acepte actualizaciones de recursos enviadas desde otros equipos (habitualmente, sus clientes DNS). Este es el sistema preferido en el caso de Windows 2000.

Por ejemplo, el servidor maestro puede admitir (e incluir en su configuración) actualizaciones de registros A y PTR de las estaciones de trabajo de su dominio, que le envían esa información cuando arrancan. También sería posible recibir estas actualizaciones de un servidor DHCP, una vez ha proporcionado la configuración IP a un cliente.

# Capítulo 3

## Niveles de ejecución en Linux

### Índice General

---

3.1. Inicio y Detención de un Sistema Linux . . . . .	21
3.2. El Concepto de Nivel de Ejecución . . . . .	21
3.3. Ficheros y Directorios de Configuración de los Niveles de Ejecución	22
3.4. Secuencia de Arranque de un Sistema Linux . . . . .	23
3.5. Secuencia de Entrada en un Nivel de Ejecución . . . . .	23
3.6. Servicios independientes de los Niveles de Ejecución . . . . .	24
3.7. Configuración de los Niveles de Ejecución . . . . .	25

---





### 3.1. Inicio y Detención de un Sistema Linux

Un sistema Linux se inicia desde el cargador del sistema (habitualmente LILO o GRUB) tecleando una simple etiqueta o realizando una selección en un menú gráfico. Un sistema Linux debe detenerse de forma correcta, nunca apagando directamente el ordenador. Para ello se utiliza el mandato `shutdown`. Este mandato se emplea habitualmente en dos situaciones:

- a) Para detener el sistema inmediatamente: `shutdown -h now`
- b) Para reiniciar el sistema inmediatamente: `shutdown -r now`

### 3.2. El Concepto de Nivel de Ejecución

Linux está programado para ejecutarse en un determinado nivel de ejecución. El número de niveles y sus nombre están predeterminados. En cambio, las acciones a realizar en cada nivel son configurables por el superusuario tal como se explica más tarde en este documento. La configuración de niveles en Redhat Linux se presenta en la Tabla 3.1 a continuación.

Nivel	Nombre	Descripción
0	<code>halt</code>	Este nivel detiene el sistema.
1	<code>single user mode</code>	Modo de administración. El sistema crea un shell con los privilegios del superusuario sin solicitar nombre de usuario o contraseña.
2	<code>multiuser</code>	Modo de funcionamiento normal sin algunos servicios de red.
3	<code>multiuser + network</code>	Como el modo 2 pero con todos los servicios de red activos, NFS por ejemplo.
4		Generalmente no utilizado
5	<code>X11</code>	El servidor X se inicia desde el primer momento.
6	<code>reboot</code>	Se reinicia el sistema.
s,S	<code>emergency single user</code>	Igual al nivel 1 pero sin acceder a los ficheros de configuración de inicio.

**Cuadro 3.1:** Niveles de ejecución en RedHat Linux.

Bajo esta perspectiva, un sistema Linux no se arranca o detiene, sino que simplemente se cambia su nivel de ejecución. Algunas consideraciones importantes sobre los niveles son:

- Durante un arranque normal, el sistema se coloca en el nivel 3 (multiusuario con

red) o en el nivel 5 (análogo al 3 pero con el sistema de ventanas activo desde el inicio).

- `shutdown -h now` cambia el nivel actual al nivel 0 (halt).
- `shutdown -r now` cambia el nivel actual al nivel 6 (reboot).
- `/sbin/telinit nivel` cambia al nivel especificado.
- `/sbin/runlevel` indica el nivel de ejecución previo y el actual.
- Desde LILO puede expresarse el nivel de ejecución deseado tras la etiqueta del sistema operativo (Linux) a cargar. Por ejemplo, para arrancar Linux en el nivel 1 se utiliza:

```
LILO boot: linux 1
```

- Igualmente, desde GRUB también puede indicarse el nivel de ejecución en el que se desea arrancar. En este caso, hay que seleccionar la entrada de Linux en el menú de GRUB y pulsar la tecla 'e' para que GRUB entre en el modo de edición. En la línea que hace referencia al kernel de Linux hay que añadir al final el nivel de ejecución al que se desea entrar. Por ejemplo, dada la siguiente línea:

```
kernel /boot/vmlinuz ro root=/dev/hdb1
```

se modificaría de esta forma:

```
kernel /boot/vmlinuz ro root=/dev/hdb1 1
```

Una vez hecha esta modificación, se debe pulsar 'b' para que GRUB arranque en el nivel indicado.

### 3.3. Ficheros y Directorios de Configuración de los Niveles de Ejecución

El administrador puede configurar las acciones que deben realizarse al entrar en un determinado nivel de ejecución. Los directorios y ficheros relevantes para esta configuración son:

- Directorio `/etc/rc.d`: En él residen todos los scripts de inicialización.
- Fichero `/etc/inittab`: Fichero base de configuración del arranque de la máquina.
- Fichero `/etc/rc.d/rc.sysinit`: Script de inicialización del ordenador, independiente del nivel.

- Directorios `/etc/rc.d/rc?.d`: Existe un directorio por cada nivel de ejecución, que contiene *enlaces simbólicos* a los scripts que configuran la entrada a este nivel.
- Directorio `/etc/rc.d/init.d`: Aquí residen todos los scripts reales que pueden ser ejecutados cuando se entra en un nivel de ejecución.

### 3.4. Secuencia de Arranque de un Sistema Linux

Cuando arranca un sistema Linux se producen las acciones siguientes:

1. Se carga e inicializa el núcleo del sistema.
2. El núcleo crea el primer proceso del sistema, denominado `init`.
3. `init` realiza las acciones especificadas en el fichero `/etc/inittab`.
4. `init` ejecuta el script `/etc/rc.d/rc.sysinit`.
5. `init` hace que el sistema entre en el nivel especificado en `/etc/inittab` o alternativamente en el nivel indicado desde el cargador (LILO o GRUB).

No es normal tener que configurar esta secuencia de arranque, dictaminada por los ficheros `/etc/inittab` y `/etc/rc.d/rc.sysinit`. Estos ficheros son configurados en origen por quien distribuye el sistema operativo. No obstante, pueden ser necesarios para configurar sistemas Linux con requerimientos muy específicos.

### 3.5. Secuencia de Entrada en un Nivel de Ejecución

Cuando se entra en un determinado nivel de ejecución, se realizan las siguientes acciones:

1. Se ejecutan, por orden de nombre, todos los scripts que comienzan por 'k' en el directorio correspondiente al nivel, utilizando como argumento para dicho script la opción `stop`.
2. Se ejecutan, por orden de nombre, todos los scripts que comienzan por 's' en el directorio correspondiente al nivel, utilizando como argumento para dicho script la opción `start`.

A título de ejemplo, a continuación se muestra un listado del directorio que corresponde al nivel multiusuario con red (`/etc/rc.d/rc3.d`).

```
ls -l rc3.d/

total 0
lrwxrwxrwx 1 root root 13 Apr 1 1998 K15gpm -> ../init.d/gpm
lrwxrwxrwx 1 root root 13 Apr 1 1998 K60lpd -> ../init.d/lpd
lrwxrwxrwx 1 root root 15 Apr 1 1998 K95nfsfs -> ../init.d/nfsfs
lrwxrwxrwx 1 root root 17 Apr 1 1998 S01kernel -> ../init.d/kernel
lrwxrwxrwx 1 root root 17 Apr 1 1998 S10network -> ../init.d/network
lrwxrwxrwx 1 root root 16 Apr 1 1998 S20random -> ../init.d/random
lrwxrwxrwx 1 root root 16 Apr 1 1998 S30syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 13 Apr 1 1998 S40atd -> ../init.d/atd
lrwxrwxrwx 1 root root 15 Apr 1 1998 S40crond -> ../init.d/crond
lrwxrwxrwx 1 root root 18 Apr 1 1998 S75keytable -> ../init.d/keytable
lrwxrwxrwx 1 root root 11 Apr 1 1998 S99local -> ../rc.local
```

Puede observarse cómo en este directorio tan sólo existen enlaces simbólicos a los verdaderos scripts que residen en el directorio `/etc/rc.d/init.d`. Cada uno de estos scripts acepta dos parámetros, `start` y `stop`, en función del cual inicia o detiene el servicio correspondiente. En este ejemplo, la secuencia de entrada al nivel 3 sería:

```
gpm stop
lpd stop
nfsfs stop
kernel start
network start
random start
syslog start
atd start
crond start
keytable start
rc.local start
```

Estos scripts que residen en el directorio `/etc/rc.d/init.d` pueden utilizarse directamente, lo que permite iniciar o detener servicios de forma manual. Por ejemplo, los siguientes mandatos detienen el subsistema de red y lo vuelven a iniciar.

```
/etc/rc.d/init.d/network stop
/etc/rc.d/init.d/network start
```

De forma alternativa, puede utilizarse la orden `service`. Los siguientes mandatos son equivalentes a los anteriores:

```
service network stop
service network start
```

### 3.6. Servicios independientes de los Niveles de Ejecución

Algunos servicios de red no se configuran tal como se ha expuesto en apartados anteriores, sino que son activados desde un servicio genérico denominado `xinetd`.

A `xinetd` se le conoce normalmente como el “super servidor de internet”, dado que puede ser configurado para atender múltiples servicios. Cuando detecta una petición para un determinado servicio, activa el programa concreto que debe atender realmente el servicio y le traslada la petición. La consecuencia de este funcionamiento es que todos los servicios dependientes de `xinetd` se activan en un determinado nivel de ejecución tan sólo si `xinetd` está activo en dicho nivel de ejecución.

El servicio `xinetd` se configura en el fichero `/etc/xinetd.conf` y en el directorio `/etc/xinetd.d`. En este directorio debe crearse un fichero de configuración por cada servicio dependiente de `xinetd`. Esta configuración es bastante compleja, pero generalmente se realiza de forma automática cuando se instalan los paquetes Red-Hat correspondientes. Para más detalles pueden consultarse la páginas de manual de `xinetd` y `xinetd.conf`.

### 3.7. Configuración de los Niveles de Ejecución

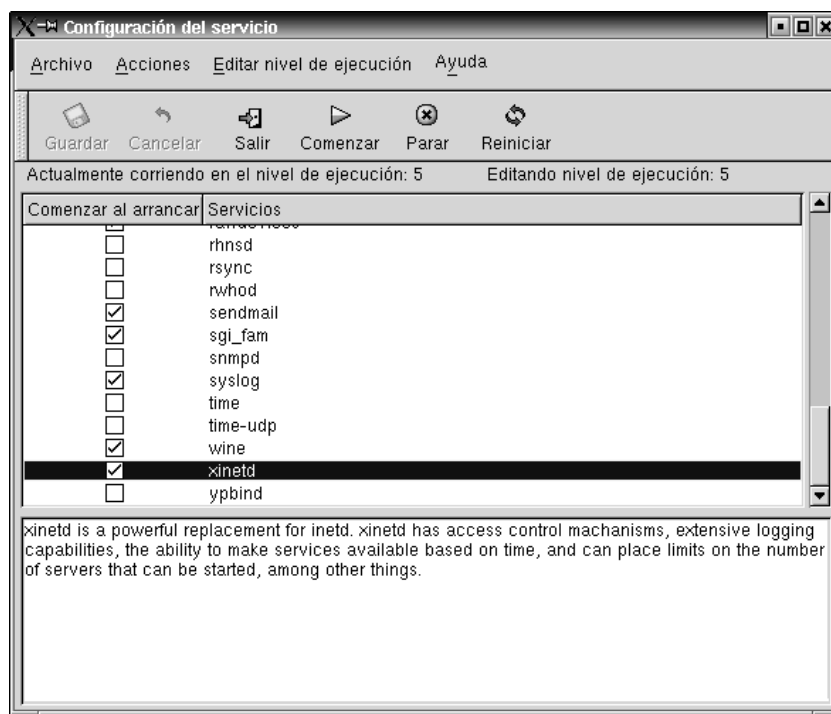
La configuración de los niveles de ejecución se limita a indicar qué scripts deben ejecutarse al entrar en cada nivel, bien para detener o para activar un determinado servicio. Para ello, basta con crear los enlaces simbólicos adecuados, o mejor todavía, utilizar el mandato `chkconfig` o alguna herramienta gráfica.

El mandato `chkconfig` permite añadir y eliminar servicios en los niveles de ejecución, así como consultar la configuración de cada servicio. La sintaxis de este mandato es la siguiente:

```
chkconfig --list [name]
chkconfig [--level levels] name <on|off|reset>
```

Utilizado con la opción `--list`, este mandato visualiza la configuración de todos los servicios o de un nivel concreto. Las acciones `on` y `off` activan y desactivan respectivamente un servicio en los niveles especificados. La acción `reset` reestablece los valores predeterminados para este servicio.

Un método alternativo consiste en utilizar una herramienta gráfica que simplifique esta labor, tal como la herramienta de configuración de servicios de RedHat, la cual permite también configurar los servicios que dependen de `xinetd` y arrancar o detener servicios manualmente. La Figura 3.1 muestra el aspecto de dicha herramienta que puede invocarse directamente mediante el mandato `/usr/bin/serviceconf`.



**Figura 3.1:** Editor de niveles de ejecución de RedHat Linux.

# Capítulo 4

## Usuarios y Protección en Linux

### Índice General

---

4.1. La Tabla de Usuarios . . . . .	29
4.2. La Extensión de la Tabla de Usuarios . . . . .	29
4.3. La Tabla de Grupos . . . . .	30
4.4. Procedimientos para la Creación de Grupos y Usuarios . . . . .	31
4.5. Atributos de Protección de los Procesos . . . . .	31
4.6. Atributos de Protección de los Ficheros . . . . .	32
4.7. Las Reglas de Protección Básicas . . . . .	33
4.8. Cambio de Atributos de Protección en Ficheros . . . . .	34
4.9. Los Bits SETUID y SETGID en Ficheros Ejecutables . . . . .	35
4.10. El Bit SETGID en Directorios . . . . .	35
4.11. La Máscara de Creación de Ficheros . . . . .	35
4.12. La Estrategia de los Grupos Privados . . . . .	36
4.13. Sintaxis y Funcionamiento de los Mandatos Unix Relacionados . .	36

---





## 4.1. La Tabla de Usuarios

Los usuarios de un sistema Unix son registrados en el fichero `/etc/passwd`. Cada línea corresponde a un usuario y describe los atributos del mismo. Los atributos son separados por el carácter `:`. Por ejemplo, dada la siguiente línea del fichero `/etc/passwd`, a continuación se describe cada elemento de la misma:

```
usul:$1$ZtoHwEyKkW/eCLHzSUigg5KAExa51:1002:2000:Usuario 1:
/home/usul:/bin/bash
```

Elemento	Significado
usul	Nombre del usuario.
\$1\$ZtoHwEyKkW/eCLHzSUigg5KAExa51	Contraseña cifrada.
1002	Identificador del usuario (UID).
2000	Identificador del grupo primario al que pertenece el usuario (GID).
Usuario 1	Descripción del usuario.
/home/usul	Directorio de conexión inicial del usuario.
/bin/bash	Programa intérprete de órdenes (shell).

Existen varios usuarios especiales predefinidos en el sistema, normalmente aquellos cuyo UID es inferior a 500. Entre estos usuarios cabe destacar a `root`, cuyo UID es igual a 0. Este es el administrador del sistema, conocido generalmente como el *superusuario*. Otros usuarios especiales sirven para asociar niveles de acceso a ciertos servicios del sistema, como por ejemplo los usuarios `mail` o `news`. Otro caso destacable es el usuario `nobody`, normalmente empleado para representar conexiones de red realizadas por usuarios desconocidos o anónimos. Todos estos usuarios se crean durante la instalación del sistema o cuando se instalan ciertos paquetes y, en principio, no deben modificarse nunca.

## 4.2. La Extensión de la Tabla de Usuarios

Además de la tabla de usuarios descrita en el apartado anterior, en la mayoría de sistemas Unix actuales se utiliza otra tabla, contenida en el fichero `/etc/shadow`, en la que se almacena información adicional para los usuarios. Cada línea de dicho fichero corresponde a una línea del fichero `/etc/passwd` y la información que contiene es la siguiente (de acuerdo con el ejemplo anterior):

```
usul:$1$ZtoHwEyKkW/eCLHzSUigg5KAExa51:10989:0:99999:7:-1:-1:134538436
```

Elemento	Significado
usu1	Nombre del usuario.
\$1\$ZtoHwEyKkW/eC LHzSUi9g5KAExa51	Contraseña cifrada.
10989:0:99999:7: -1:-1:134538436	Información de caducidad de la contraseña.

Cuando el sistema emplea esta tabla, la contraseña que debería estar almacenada en el fichero `/etc/passwd` se sustituye por una 'x'. Dado que el fichero `/etc/passwd` es legible por todos los usuarios mientras que `/etc/shadow` sólo es legible por el administrador, mediante este cambio se consigue que ningún usuario tenga acceso a las contraseñas cifradas.

La información de caducidad de la contraseña está expresada en días, siendo su utilización e interpretación compleja en algunos casos. Por este motivo, para modificar y consultar esta información no se recomienda en general trabajar directamente con el fichero `/etc/shadow`, sino utilizar herramientas administrativas específicas, ya sea en forma de mandatos (`passwd` y `chage`) o mediante aplicaciones gráficas.

### 4.3. La Tabla de Grupos

Los grupos de un sistema Unix son registrados en un fichero denominado `/etc/group`. Cada línea corresponde a un grupo y describe los atributos del mismo. Los atributos son separados por el carácter ':' y su significado se describe mediante el siguiente ejemplo:

```
proy1::65534:usu1,usu2,usu3
```

Elemento	Significado
proy1	Nombre del grupo.  Contraseña cifrada (vacía en el ejemplo). Esta contraseña permite a un usuario cambiar de grupo primario si se conoce esta contraseña. Esta funcionalidad está en desuso.
65534	Identificador del grupo (GID).
usu1,usu2,usu3	Lista de usuarios que pertenecen a este grupo.

Como se deduce del ejemplo, un usuario puede figurar en la lista de distintos grupos, es decir, puede pertenecer a *varios* grupos. De todos esos grupos, aquél cuyo GID figura en la entrada correspondiente al usuario en el fichero `/etc/passwd` se considera el *grupo primario* de dicho usuario. El resto de grupos se denominan *grupos suplementarios*.

Al igual que sucede con los usuarios, existen ciertos grupos que son especiales, en concreto aquellos cuyo GID es menor que 500. De forma análoga a los usuarios especiales, representan servicios, usuarios anónimos, etc. No deben, por tanto, ser modificados.

## 4.4. Procedimientos para la Creación de Grupos y Usuarios

En la mayoría de sistemas Unix System V pueden emplearse los siguientes mandatos para la gestión de usuarios y grupos:

Mandato	Uso
<code>useradd</code>	Añadir un usuario.
<code>userdel</code>	Eliminar un usuario.
<code>usermod</code>	Modificar los atributos de un usuario.
<code>groupadd</code>	Añadir un grupo.
<code>groupdel</code>	Eliminar un grupo.
<code>groupmod</code>	Modificar los atributos de un grupo.
<code>passwd</code>	Cambiar la contraseña de un usuario.
<code>chage</code>	Gestionar la caducidad de la contraseña.

Estos mandatos son especialmente útiles cuando la gestión de usuarios debe realizarse desde *scripts* del *shell*, por ejemplo, cuando debe crearse una gran cantidad de usuarios de forma automática. La sintaxis y opciones más usuales de estos mandatos se citan en la Sección 4.13, al final de este capítulo.

Adicionalmente, cada vez es más frecuente la disponibilidad de herramientas gráficas que facilitan la labor del administrador, tal como la utilidad “User Manager” de RedHat.

## 4.5. Atributos de Protección de los Procesos

Los atributos de un proceso que intervienen en el mecanismo de protección son:

- a) **Identificadores del usuario propietario del proceso (rUID, eUID).** La versión real —rUID— corresponde al usuario que creó al proceso. La versión efectiva —eUID— corresponde al usuario bajo el cual se comporta el proceso y es el utilizado en el mecanismo de protección. Ambos identificadores suelen ser iguales, salvo que intervenga el denominado bit SETUID en el fichero ejecutable que está ejecutando el proceso, tal como se describe más adelante en este documento (ver Sección 4.9).

- b) **Identificadores del grupo propietario del proceso (rGID, eGID).** La versión real —rGID— corresponde al grupo primario al que pertenece el usuario que creó el proceso. La versión efectiva —eGID— corresponde al grupo bajo el cual se comporta el proceso y es el utilizado en el mecanismo de protección. Ambos identificadores suelen ser iguales, salvo que intervenga el denominado bit SETGID en el fichero ejecutable que está ejecutando el proceso, tal como se describe más adelante en este documento (ver Sección 4.9).
- c) **Lista de grupos suplementarios.** Es la lista formada por los grupos suplementarios del usuario que creó el proceso.

Estos atributos son asignados al proceso en el momento de su creación, y heredados de su proceso padre. Cada usuario conectado a un sistema Unix posee un proceso de atención, el cual puede ser un entorno de ventanas o un intérprete de órdenes (*shell*). En cualquiera de ambos casos, este proceso es el padre de todos los procesos que el usuario genera. Este proceso shell recibe sus atributos no por herencia, sino en el momento en el que un usuario se acredita al sistema mediante su nombre de usuario y contraseña asociada. Los atributos rUID y rGID se extraen de la tabla de usuarios `/etc/passwd`. La lista de grupos suplementarios es confeccionada a partir de la tabla de grupos `/etc/group`. Los atributos eUID y eGID son asignados a los valores respectivos de rUID y rGID. Un mandato interesante al respecto es `id`, el cual muestra todos estos atributos.

## 4.6. Atributos de Protección de los Ficheros

Los atributos de un fichero que intervienen en el mecanismo de protección son:

- a) **OwnerUID.** Identificador del usuario propietario del fichero.
- b) **OwnerGID.** Identificador del grupo propietario del fichero.
- c) **Bits de permiso.** Un total de 12 bits que expresan las operaciones del fichero que son permitidas en función del proceso que acceda a este fichero.

En la Tabla 4.6 se muestra el significado de cada bit de permiso.

El significado de los bits de lectura, escritura y ejecución es diferente en función del tipo de archivo que los defina. Para ficheros regulares, su significado es el esperable (permiten leer, modificar y ejecutar el fichero respectivamente). Evidentemente, el bit de ejecución sólo tiene sentido si el fichero es un ejecutable o contiene un *shellscript*. En un directorio, su significado es el siguiente:

- a) **Lectura.** Puede listarse el contenido del directorio.
- b) **Escritura.** Permite la creación, eliminación o renombrado de ficheros o directorios dentro del directorio al cual se aplica este bit.

Bit	Significado
11	SETUID.
10	SETGID.
9	Sticky.
8	Lectura para el propietario.
7	Escritura para el propietario.
6	Ejecución para el propietario.
5	Lectura para el grupo propietario.
4	Escritura para el grupo propietario.
3	Ejecución para el grupo propietario.
2	Lectura para el resto de usuarios.
1	Escritura para el resto de usuarios.
0	Ejecución para el resto de usuarios.

**Cuadro 4.1:** Significado de los bits de permiso en Unix.

- c) **Ejecución.** Permite la utilización del directorio al cual se aplica este bit para formar parte de un nombre de ruta, es decir, puede utilizarse el directorio para nombrar un fichero.

De lo anteriormente expuesto, puede observarse que no existe un bit de permiso específico para el borrado de un fichero/directorio. Dicho permiso es controlado realmente desde el directorio donde reside el fichero que quiere eliminarse. En algunos sistemas Unix (como por ejemplo RedHat Linux), el bit sticky es utilizado para modificar la regla de eliminación de ficheros: si se activa en un directorio, un usuario puede borrar un fichero en él sólo si es el propietario de dicho fichero. Los bits SETUID y SETGID son tratados en apartados posteriores.

El ownerUID puede modificarse con el mandato `chown`. El ownerGID puede modificarse con el mandato `chgrp`. Los bits de permiso pueden modificarse con el mandato `chmod`. Todos estos mandatos se describen en la Sección 4.13.

## 4.7. Las Reglas de Protección Básicas

Las reglas de protección básicas se activan cuando un proceso notifica al sistema que desea utilizar un determinado fichero. El proceso también notifica al sistema qué tipo de operaciones quiere realizar: lectura, escritura o ejecución.

Las reglas son:

- Si el eUID del proceso es 0 se concede el permiso (estamos en el caso en el que el proceso pertenece al superusuario). Si no...

- Si el eUID del proceso es igual al ownerUID del fichero, se autoriza la operación si está permitida en el grupo de bits 6 al 8, los que corresponden al propietario. Si no...
- Si el eGID del proceso o alguno de los grupos suplementarios del proceso es igual al ownerGID del fichero, se autoriza la operación si está permitida en el grupo de bits 3 al 5, los que corresponden al grupo. Si no...
- En cualquier otro caso, se autoriza la operación si está permitida en el grupo de bits 0 al 2, los que corresponden al resto de usuarios.

Debe notarse que sólo se aplica una regla, aquella que corresponde a la primera condición que se cumple para los atributos del proceso. Es decir, el sistema determina *primero* qué grupo de tres bits debe aplicar y *después* autoriza (o deniega) la operación en función del tipo de operación requerida y del estado de dichos tres bits.

## 4.8. Cambio de Atributos de Protección en Ficheros

Unix establece unas reglas de protección específicas para controlar los cambios de cualquier atributo de protección de un fichero, dado que dichas modificaciones se consideran operaciones *distintas* de la de escritura, y por tanto no pueden ser concedidas/denegadas en función de los bits de permiso del fichero.

En concreto, las reglas establecidas al respecto son las siguientes:

- a) **Cambio en los bits de permiso.** Un proceso puede cambiar los bits de permiso de un fichero sólo si:
- el eUID del proceso es 0 (estamos en el caso en el que el proceso pertenece al superusuario), o bien
  - el eUID del proceso es igual al ownerUID del fichero.

Es decir, sólo el superusuario o el propietario de un fichero pueden modificar sus bits de permiso.

- b) **Cambio en el propietario.** El cambio de propietario de un fichero puede realizarlo tan sólo el superusuario.
- c) **Cambio de grupo propietario.** El cambio del grupo propietario de un fichero puede realizarse sólo si:
- lo realiza el superusuario, o bien
  - lo realiza el propietario del fichero, siempre que el nuevo ownerGID del fichero sea igual a alguno de los grupos de dicho usuario.

## 4.9. Los Bits SETUID y SETGID en Ficheros Ejecutables

Estos bits se emplean para permitir que un programa se ejecute bajo los privilegios de un usuario distinto al que lanza la ejecución del programa. Su funcionamiento es el siguiente:

- Si el fichero ejecutable tiene el bit SETUID activo, el eUID del proceso que ejecuta el fichero es hecho igual al ownerUID del fichero.
- Si el fichero ejecutable tiene el bit SETGID activo, el eGID del proceso que ejecuta el fichero es hecho igual al ownerGID del fichero.

Normalmente estos programas pertenecen al superusuario, y permiten a los usuarios ejecutar tareas privilegiadas bajo ciertas condiciones. El cambio de la contraseña de un usuario es un ejemplo de esta técnica.

La existencia de estos ficheros en el sistema de archivos debe ser cuidadosamente supervisada por el superusuario. Muchos de los ataques de seguridad a Unix utilizan estos ficheros para conseguir atentar contra la seguridad del sistema.

## 4.10. El Bit SETGID en Directorios

La utilización de este bit está orientada a facilitar el trabajo en grupo cuando varios usuarios deben acceder a una colección común de ficheros y directorios. Su funcionamiento es el siguiente: si un directorio, llamémosle *D*, tiene el bit SETGID activo, entonces:

- si se crea un fichero dentro de *D*, el ownerGID del nuevo fichero es hecho igual al ownerGID del directorio *D*.
- si se crea un directorio dentro de *D*, el ownerGID del nuevo directorio es hecho igual al ownerGID del directorio *D* y su bit SETGID es activado.

Puede observarse que se trata, en esencia, de un mecanismo de herencia. Cuando el bit SETGID no está activo, el ownerGID de un fichero o directorio se toma del eGID del proceso que lo crea. En cambio, utilizando el SETGID, aunque varios usuarios creen ficheros dentro de un directorio, todos ellos pertenecerán al menos al mismo grupo, con lo que una utilización adecuada de los bits de permiso puede hacer que todos ellos puedan, por ejemplo, leer y modificar dichos ficheros.

## 4.11. La Máscara de Creación de Ficheros

Las utilidades de Unix que crean ficheros utilizan por defecto la palabra de protección `rw-rw-rw` si se trata de ficheros no ejecutables o `rw-rw-rw-rw` si se crea un

directorio o un ejecutable. Puede observarse, por tanto, que todos los permisos son activados.

Para modificar este funcionamiento, cada usuario puede especificar al sistema aquellos bits que no desea que sean activados, mediante la máscara de creación de ficheros. La máscara se establece mediante el mandato `umask`. Cada bit activo en la máscara es un bit que será desactivado cuando se cree un fichero. Por ejemplo, una máscara 022 desactivaría los bits de escritura para el grupo y el resto de usuarios.

El administrador debe velar porque exista una máscara de creación de ficheros razonable por defecto para cualquier usuario. Esto puede conseguirse fácilmente utilizando el fichero `/etc/profile`, el cual sirve para personalizar el entorno de los usuarios en el momento de su conexión.

## 4.12. La Estrategia de los Grupos Privados

Esta estrategia, empleada por defecto en Redhat Linux, está orientada a racionalizar y flexibilizar la asignación de usuarios a grupos. Las claves de esta estrategia son:

- Crear un grupo privado para cada usuario (utilizando para ello el mismo nombre del usuario), y hacer que éste sea el grupo primario del usuario.
- Utilizar exclusivamente grupos suplementarios para agrupar usuarios.
- Utilizar el bit SETGID y un grupo suplementario en aquellos directorios donde varios usuarios tengan que colaborar.
- Utilizar el grupo privado en el directorio de conexión de cada usuario.
- Asignar como máscara por defecto para todos los usuarios 00X, es decir, todos los permisos activos para el propietario y para el grupo y lo que se considere oportuno para el resto de usuarios.

Se invita al lector a reflexionar sobre las consecuencias de esta estrategia.

## 4.13. Sintaxis y Funcionamiento de los Mandatos Unix Relacionados

```

useradd
useradd [-u uid [-o]] [-g group] [-G group,...]
        [-d home] [-s shell] [-c comment] [-m [-k template]]
        [-f inactive] [-e expire mm/dd/yy] [-p passwd] [-n] [-r] name
useradd -D [-g group] [-b base] [-s shell] [-f inactive]
        [-e expire mm/dd/yy]
```



Crea un usuario. Opciones:

- c Comentario sobre el usuario
- d Directorio de conexión del usuario
- e Fecha en la cual la cuenta de usuario se desactiva
- f Días que transcurrirán desde la caducidad de la contraseña hasta la desactivación de la cuenta
- g Nombre o número del grupo primario
- G Lista de los grupos suplementarios del usuario (separados por ,) (sin espacios)
- m Crea el directorio de conexión. Si no se especifica la opción -k, copia al directorio de conexión los ficheros del directorio /etc/skel
- k Copia al directorio de conexión los ficheros del directorio template
- p Asigna una contraseña cifrada al usuario
- r Crea una cuenta del sistema
- s Asigna el shell a utilizar por el usuario
- u Asigna un UID concreto al usuario
- o Permite crear un UID duplicado con la opción -u
- D Se asignan valores por defecto para las opciones indicadas. (No crea usuario)

-----

```
userdel
userdel [-r] name
```

Elimina un usuario. Opciones:

- r Elimina el directorio de conexión del usuario

-----

```
usermod
usermod [-u uid [-o]] [-g group] [-G group,...]
        [-d home [-m]] [-s shell] [-c comment] [-l new_name]
        [-f inactive] [-e expire mm/dd/yy] [-p passwd] [-L|-U] name
```

Modifica atributos de un usuario. Opciones:

- c Comentario sobre el usuario
- d Directorio de conexión del usuario
- e Fecha en la cual la cuenta de usuario se desactiva
- f Días que transcurrirán desde la caducidad de la contraseña hasta la desactivación de la cuenta
- g Nombre o número del grupo primario
- G Lista de los grupos suplementarios del usuario (separados por ,) (sin espacios)
- l Modifica el nombre de conexión del usuario
- L Bloquea la contraseña del usuario
- m Crea el directorio de conexión Si no se especifica la opción -k, copia al directorio de conexión los ficheros del directorio /etc/skel
- p Asigna una contraseña cifrada al usuario
- s Asigna el shell a utilizar por el usuario
- u Asigna un UID concreto al usuario
- o Permite crear un UID duplicado con la opción -u
- U Desbloquea la contraseña del usuario

---

```
groupadd
groupadd [-g gid [-o]] [-r] group
```

Crea un grupo de usuarios. Opciones:

```
-g Asigna un GID concreto al grupo
-o Permite crear un GID duplicado con la opción -g
-r Crea una cuenta de grupo del sistema
```

---

```
groupmod
groupmod [-g gid [-o]] [-n name] group
```

Modifica los atributos de un grupo de usuarios. Opciones:

```
-g Asigna un GID concreto al grupo
-o Permite crear un GID duplicado con la opción -g
-n Cambia el nombre del grupo
```

---

```
groupdel
groupdel group
```

Elimina un grupo de usuarios. No tiene opciones.

---

```
passwd
passwd [-l] [-u [-f]] [-d] [-S] [ username ]
```

Modifica la contraseña y el estado de la cuenta un usuario. Opciones:

```
-d Elimina la contraseña del usuario
-f Fuerza el desbloqueo
-l Bloquea la cuenta del usuario
-S Informa sobre el estado de la cuenta de un usuario
-u Desbloquea la cuenta del usuario
```

Sin argumentos modifica la contraseña del usuario que lo invoca.  
Si se especifica un nombre de usuario y no se especifican opciones,  
modifica la contraseña del usuario indicado

---

```
chage
chage [ -l ] [ -m min_days ] [ -M max_days ] [ -W warn ]
      [ -I inactive ] [ -E expire ] [ -d last_day ] user
```

Modifica la información de caducidad de la contraseña de un usuario.  
Opciones:

```
-d Asigna la fecha del último cambio de contraseña
-E Asigna la fecha en la cual la cuenta de usuario será desactivada
-l Muestra la información de caducidad
-I Número de días tras los cuales la cuenta será desactivada si no
  se realiza el cambio de contraseña exigido
-m Número de días mínimo permitido entre cambios de contraseña
```

-M Número de días máximo permitido entre cambios de contraseña  
 -W Número de días previos al próximo cambio de contraseña exigido

---

chown  
 chwon [ -R ] owner file

Modifica el usuario propietario de un fichero o directorio. Opciones:

-R Realiza la modificación en todo el contenido del directorio

---

chgrp  
 chgrp [ -R ] group file

Modifica el grupo propietario de un fichero o directorio. Opciones:

-R Realiza la modificación en todo el contenido del directorio

---

chmod  
 chmod -R MODE[,MODE]... FILE...  
 chmod -R OCTAL\_MODE FILE...

Modifica los bits de permiso de un fichero o directorio. Opciones:

-R Realiza la modificación en todo el contenido del directorio

MODE indica como deben modificarse los bits de permiso. Su sintaxis es:

[ugoa][+ -=][rwxXstugo]

[ugoa]            u propietario    g grupo            o otros            a todos

[+ -=]            + añadir            - eliminar            = hacer igual

[rwxXstugo]      r lectura  
                   w escritura  
                   x ejecución  
                   X ejecución si activo para el propietario  
                   s SETUID o SETGID  
                   t sticky  
                   u igual al propietario  
                   g igual al grupo  
                   o igual a otros

---



# Capítulo 5

## Protección Local en Windows 2000

### Índice General

---

<b>5.1. Conceptos de Usuario y de Cuenta de Usuario . . . . .</b>	<b>43</b>
<b>5.2. Grupos de Usuarios . . . . .</b>	<b>44</b>
<b>5.3. El Modelo de Protección de Windows 2000 . . . . .</b>	<b>45</b>
<b>5.4. Atributos de Protección de los Procesos . . . . .</b>	<b>46</b>
<b>5.5. Derechos de Usuario . . . . .</b>	<b>47</b>
5.5.1. Otras Directivas de Seguridad . . . . .	48
<b>5.6. Atributos de Protección de los Recursos . . . . .</b>	<b>48</b>
5.6.1. Permisos Estándar y Permisos Individuales . . . . .	50
<b>5.7. Reglas de Protección . . . . .</b>	<b>51</b>

---



## 5.1. Conceptos de Usuario y de Cuenta de Usuario

Como muchos otros sistemas operativos, Windows 2000 permite tener un riguroso control de las personas que pueden entrar en el sistema y de las acciones que dichas personas están autorizadas a ejecutar.

Windows 2000 denomina *usuario* a cada persona que puede entrar en el sistema. Para poder controlar la entrada y las acciones de cada usuario utiliza básicamente el concepto de *cuenta de usuario* (user account). Una cuenta de usuario almacena toda la información que el sistema guarda acerca de cada usuario. De entre los numerosos datos que Windows 2000 almacena en cada cuenta de usuario, los más importantes son los siguientes:

- a) **Nombre de usuario.** Es el nombre mediante el cual el usuario *se identifica* en el sistema. Cada usuario ha de tener un nombre de usuario distinto para que la identificación sea unívoca.
- b) **Nombre completo.** Es el nombre completo del usuario.
- c) **Contraseña.** Palabra cifrada que permite *autenticar* el nombre de usuario. En Windows 2000 la contraseña distingue entre mayúsculas y minúsculas. Sólo los usuarios que se identifican y autentican positivamente pueden ser *autorizados* a conectarse al sistema.
- d) **Directorio de conexión.** Es el lugar donde (en principio) residirán los archivos personales del usuario. El directorio de conexión de cada usuario es privado: ningún otro usuario puede entrar en él, a menos que su propietario conceda los permisos adecuados.
- e) **Horas de conexión.** Se puede controlar a qué horas un usuario puede conectarse para trabajar en el sistema. Inclusive se puede especificar un horario distinto para cada día de la semana.
- f) **Activada.** Esta característica permite inhabilitar temporalmente una cuenta. Una cuenta desactivada sigue existiendo, pero no puede ser utilizada para acceder al sistema, ni siquiera conociendo su contraseña.

Existe un dato especial que se asocia a cada cuenta, pero que a diferencia de todos los expuestos arriba, no puede ser especificado manualmente cuando se da de alta la cuenta. Se trata del *identificador seguro* (Secure Identifier, o SID). Este identificador es interno y el sistema lo genera automáticamente cuando se crea una nueva cuenta. Además, los SIDs se generan de tal forma que se asegura que no pueden existir dos iguales en todas las instalaciones de Windows 2000 del mundo (son identificadores únicos). Windows 2000 utiliza siempre el SID (y no el nombre de usuario) para controlar si un usuario tiene o no permisos suficientes para llevar a cabo cualquiera de sus acciones. La ventaja de este modelo es que el SID es un dato completamente interno del sistema operativo, es decir, ningún usuario puede establecerlo en ningún sitio

(ni siquiera el administrador del sistema). Por tanto, nadie puede obtener un mayor grado de privilegio intentando *suplantar* la identidad de otro usuario.

Cuando en un equipo se instala Windows 2000, existen de entrada las cuentas de dos usuarios preinstalados (built-in users): el Administrador y el Invitado. El primero es un usuario especial, el único que en principio posee lo que se denominan derechos administrativos en el sistema. Es decir, tiene la potestad de administrar el sistema en todos aquellos aspectos en que éste es configurable: usuarios, grupos de usuarios, contraseñas, recursos, derechos, etc. La cuenta de Administrador no puede ser borrada ni desactivada. Por su parte, la cuenta de Invitado es la que utilizan normalmente aquellas personas que no tienen un usuario propio para acceder al sistema. Habitualmente esta cuenta no tiene contraseña asignada, puesto que se supone que el nivel de privilegios asociado a ella es mínimo. En cualquier caso, el Administrador puede desactivarla si lo considera oportuno.

## 5.2. Grupos de Usuarios

La información de seguridad almacenada en una cuenta de usuario es suficiente para establecer el grado de libertad (o de otro modo, las restricciones) que cada usuario debe poseer en el sistema. Sin embargo, resultaría muchas veces tedioso para el administrador determinar dichas restricciones usuario por usuario, especialmente en sistemas con un elevado número de ellos. El concepto de *grupo de usuarios* permite agrupar de forma lógica a los usuarios de un sistema, y establecer permisos y restricciones a todo el grupo de una vez. Un usuario puede pertenecer a tantos grupos como sea necesario, poseyendo implícitamente la *suma* de los permisos de todos ellos. Esta forma de administrar la protección del sistema es mucho más flexible y potente que el establecimiento de permisos en base a usuarios individuales.

Considérese, por ejemplo, que en una empresa un sistema es utilizado por empleados de distinto rango, y que cada rango posee un distinto nivel de privilegios. Supongamos que se desea cambiar de rango a un empleado, debido a un ascenso, por ejemplo. Si la seguridad estuviera basada en usuarios individuales, cambiar los privilegios de este usuario adecuadamente supondría modificar sus permisos en cada lugar del sistema en que estos debieran cambiar (con el consiguiente trabajo, y el riesgo de olvidar alguno). Por el contrario, con la administración de seguridad basada en grupos, esta operación sería tan sencilla como cambiar al usuario de un grupo a otro. Por ello, en Windows 2000 se recomienda que los permisos se asignen en base a *grupos*, y no en base a usuarios individuales.

Al igual que existen cuentas preinstaladas de usuario, en todo sistema 2000 existen una serie de grupos preinstalados (built-in groups): Administradores, Operadores de Copia, Usuarios Avanzados, Usuarios, e Invitados. El grupo Administradores recoge a todos aquellos usuarios que deban poseer derechos administrativos completos. Inicialmente posee un solo usuario, el Administrador. De igual forma, el grupo Invitados posee al Invitado como único miembro. Los otros tres grupos están vacíos inicialmente. Su uso es el siguiente:



- a) **Usuarios.** Son los usuarios normales del sistema. Tienen permisos para conectarse al sistema interactivamente y a través de la red.
- b) **Operadores de copia.** Estos usuarios pueden hacer (y restaurar) copias de todo el sistema.
- c) **Usuarios avanzados.** Son usuarios con una cierta capacidad administrativa. Se les permite cambiar la hora del sistema, crear cuentas de usuario y grupos, compartir ficheros e impresoras, etc.

El Administrador, al ir creando las cuentas de los usuarios, puede hacer que cada una pertenezca al grupo (o grupos) que estime conveniente. Asimismo, puede crear nuevos grupos que refinan esta estructura inicial, conforme a las necesidades particulares de la organización donde se ubique el sistema.

Finalmente, Windows 2000 define una serie de grupos especiales, cuyos (usuarios) miembros no se establecen de forma manual, sino que son determinados de forma dinámica y automática por el sistema. Estos grupos se utilizan normalmente para facilitar la labor de establecer la protección del sistema. De entre estos grupos, destacan:

- a) **Usuarios Interactivos** (Interactive). Este grupo representa a todos aquellos usuarios que tienen el derecho de iniciar una sesión local en la máquina.
- b) **Usuarios de Red** (Network). Bajo este nombre se agrupa a todos aquellos usuarios que tienen el derecho de acceder al equipo desde la red.
- c) **Todos** (Everyone). Agrupa a todos los usuarios que el sistema conoce. Puede incluir usuarios existentes localmente y de otros sistemas (conectados a través de la red).
- d) **Usuarios Autenticados.** Formado por todos los usuarios que acceden al sistema tras ser acreditados. Equivale al grupo **Todos** exceptuando al usuario **Invitado**.

### 5.3. El Modelo de Protección de Windows 2000

El modelo de protección de Windows 2000 establece la forma en que el sistema lleva a cabo el *control de acceso* de cada usuario y grupo de usuarios. En otras palabras, es el modelo que sigue el sistema para establecer las acciones que un usuario (o grupo) está autorizado a llevar a cabo. Este modelo está basado en la definición y contrastación de ciertos *atributos de protección* que se asignan a los procesos de usuario por un lado, y al sistema y sus recursos por otro. En el caso del sistema y sus recursos, Windows 2000 define respectivamente dos conceptos distintos y complementarios: el concepto de *derecho* y el concepto de *permiso*.

Un *derecho o privilegio* de usuario (user right) es un atributo de un usuario (o grupo) que le permite realizar una acción que afecta al sistema en su conjunto (y no a un objeto o recurso en concreto). Existe un conjunto fijo y predefinido de derechos en Windows

2000. Para determinar qué usuarios poseen qué derechos, cada derecho posee una lista donde se especifican los grupos/usuarios que tienen concedido este derecho.

Un *permiso* (permission) es una característica de cada *recurso* (carpeta, archivo, impresora, etc.) del sistema, que concede o deniega el acceso al mismo a un usuario/grupo concreto. Cada recurso del sistema posee una lista en la que se establece qué usuarios/grupos pueden acceder a dicho recurso, y también qué tipo de acceso puede hacer cada uno (lectura, modificación, ejecución, borrado, etc.).

En los apartados siguientes se detallan los atributos de protección de los procesos de usuario (Apartado 5.4), los derechos que pueden establecerse en el sistema (Apartado 5.5) y los atributos de protección que poseen los recursos (Apartado 5.6). El Apartado 5.7 establece las reglas concretas que definen el control de acceso de los procesos a los recursos.

## 5.4. Atributos de Protección de los Procesos

Cuando un usuario es autorizado a conectarse interactivamente a un sistema Windows 2000, el sistema construye para él una acreditación denominada *Security Access Token* o SAT. Esta acreditación contiene la información de protección del usuario, y Windows 2000 la incluye en los procesos que crea para dicho usuario. De esta forma, los *atributos de protección* del usuario están presentes en cada proceso del usuario, y se utilizan para controlar los accesos que el proceso realiza a los recursos del sistema en nombre de dicho usuario.

En concreto, el SAT contiene los siguientes atributos de protección:

- a) **SID del usuario.**
- b) **Lista de SIDs de los grupos a los que pertenece el usuario.**
- c) **Lista de derechos del usuario.** Esta lista se construye mediante la inclusión de todos los derechos que el usuario tiene otorgados por sí mismo o por los grupos a los que pertenece (ver Apartado 5.5).

Esta forma de construir la acreditación introduce ya una de las máximas de la protección de Windows 2000: el nivel de acceso de un usuario incluye implícitamente los niveles de los grupos a los que pertenece.

Windows 2000 incorpora la posibilidad de ejecutar programas como un usuario distinto. Para ello, desde el entorno de ventanas basta con pinchar el icono del programa con el botón derecho del ratón mientras se mantiene pulsada la tecla Mayúsculas. En el menú contextual que aparece se debe seleccionar la opción *Ejecutar como...*. En línea de órdenes se puede realizar la misma acción utilizando el mandato `RUNAS`, tal como se muestra a continuación.

```
RUNAS /USER:dominio_o_máquina\usuario programa
```

## 5.5. Derechos de Usuario

Un *derecho* es un atributo de un usuario o grupo de usuarios que le confiere la posibilidad de realizar una acción concreta sobre el sistema en conjunto (no sobre un recurso concreto). Como hemos visto, la lista de derechos de cada usuario se añade explícitamente a la acreditación (SAT) que el sistema construye cuando el usuario se conecta al sistema. Esta lista incluye los derechos que el usuario tiene concedidos a título individual más los que tienen concedidos todos los grupos a los que el usuario pertenece.

DERECHOS DE CONEXIÓN	
Nombre	Significado
Acceder a este equipo desde la red	Permite/impide al usuario conectar con el ordenador desde otro ordenador a través de la red.
Conectarse localmente	Permite/impide al usuario iniciar una sesión local en el ordenador, desde el teclado del mismo.
PRIVILEGIOS	
Nombre	Significado
Añadir estaciones al dominio	Permite al usuario añadir ordenadores al dominio actual.
Hacer copias de seguridad	Permite al usuario hacer copias de seguridad de archivos y carpetas.
Restaurar copias de seguridad	Permite al usuario restaurar copias de seguridad de archivos y carpetas.
Atravesar carpetas	Permite al usuario acceder a archivos a los que tiene permisos a través de una ruta de directorios en los que puede no tener ningún permiso.
Cambiar la hora del sistema	Permite al usuario modificar la hora interna del ordenador.
Instalar manejadores de dispositivo	Permite al usuario instalar y desinstalar manejadores de dispositivos "Plug and Play".
Apagar el sistema	Permite al usuario apagar el ordenador local.
Tomar posesión de archivos y otros objetos	Permite al usuario tomar posesión (hacerse propietario) de cualquier objeto con atributos de seguridad del sistema (archivos, carpetas, objetos del Directorio Activo, etc.).

**Cuadro 5.1:** Derechos más importantes en Windows 2000

Windows 2000 distingue entre dos tipos de derechos: los *derechos de conexión* (login rights) y los *privilegios* (privileges). Los primeros establecen las diferentes formas en que un usuario puede conectarse al sistema (de forma interactiva, a través de la red, etc.), mientras que los segundos hacen referencia a ciertas acciones predefinidas que el usuario puede realizar una vez conectado al sistema. La Tabla 5.1 presenta los derechos más destacados de cada tipo, junto con su descripción.

Es importante hacer notar lo siguiente: cuando existe un conflicto entre lo que

concede o deniega un permiso y lo que concede o deniega un derecho, este último tiene prioridad. Por ejemplo: los miembros del grupo Operadores de Copia poseen el derecho de realizar una copia de seguridad de todos los archivos del sistema. Es posible (y muy probable) que existan archivos sobre los que no tengan ningún tipo de permiso. Sin embargo, al ser el derecho más prioritario, podrán realizar la copia sin problemas. De igual forma, el administrador tiene el derecho de tomar posesión de cualquier archivo, inclusive de aquellos archivos sobre los que no tenga ningún permiso. Es decir, como regla general, los derechos y privilegios siempre prevalecen ante los permisos particulares de un objeto, en caso de que haya conflicto.

### 5.5.1. Otras Directivas de Seguridad

En Windows 2000, los derechos son un tipo de **directivas de seguridad**. En este sentido, Windows 2000 ha agrupado un conjunto de reglas de seguridad que en NT 4.0 estaban dispersas en distintas herramientas administrativas, y las ha incorporado a una consola de administración con ese nombre (*directivas de seguridad local*).

Dentro de esta herramienta de administración podemos establecer, entre otras, los siguientes tipos de reglas de seguridad para el equipo local:

- a) **Cuentas.** En este apartado podemos establecer cuál es la *política de cuentas* o de contraseñas que sigue el equipo para sus usuarios locales. Dentro de este apartado se pueden distinguir reglas en tres epígrafes: *Contraseñas*, *Bloqueo* y *Kerberos*. Entre ellas, las dos primeras hacen referencia a cómo deben ser las contraseñas en el equipo (longitud mínima, vigencia máxima, historial, etc.) y cómo se debe bloquear una cuenta que haya alcanzado un cierto máximo de intentos fallidos de conexión local.
- b) **Directiva local.** Dentro de este apartado se encuentra, por una parte, la *Auditoría* del equipo, que permite registrar en el visor de sucesos ciertos eventos que sean interesantes, a criterio del administrador (por ejemplo, inicios de sesión local). Por otra parte, este apartado incluye los *Derechos y Privilegios* que acabamos de explicar.
- c) **Claves públicas.** Este apartado permite administrar las opciones de seguridad de las claves públicas emitidas por el equipo.

## 5.6. Atributos de Protección de los Recursos

En Windows 2000, cada carpeta o archivo posee los siguientes atributos de protección:

- a) **SID del usuario propietario.** Inicialmente, el propietario es siempre el usuario que ha creado el archivo o carpeta, aunque este atributo puede ser luego modificado bajo ciertas reglas (esto se explica más adelante).

- b) **Lista de control de acceso de protección.** Esta lista incluye los **permisos** que los usuarios tienen sobre el archivo o carpeta. La lista puede contener un número indefinido de entradas, de forma que cada una de ellas concede o deniega un conjunto concreto de permisos a un usuario o grupo conocido por el sistema. Por tanto, Windows 2000 permite definir multitud de niveles de acceso a cada objeto del sistema de archivos, cada uno de los cuales puede ser *positivo* (se otorga un permiso) o *negativo* (se deniega un permiso).
- c) **Lista de control de acceso de seguridad.** Esta segunda lista se utiliza para definir qué acciones sobre un archivo o carpeta tiene que **auditar** el sistema. El proceso de auditoría supone la anotación en el *registro del sistema* de las acciones que los usuarios realizan sobre archivos o carpetas. El sistema sólo audita las acciones especificadas (de los usuarios o grupos especificados) en la lista de seguridad de cada archivo o carpeta. Esta lista está inicialmente vacía en todos los objetos del sistema de archivos.

Cada una de dichas listas de control de acceso se divide realmente en dos listas, cada una de ellas denominada *Discretionary Access Control List* (lista de control de acceso discrecional) o DACL. Cada elemento de una DACL se denomina *Access Control Entry* (entrada de control de acceso) o ACE, y liga un SID de usuario o grupo con la concesión o denegación de un permiso concreto (o conjunto de permisos), tal como se ha descrito arriba. Los diferentes permisos que se pueden asignar a usuarios o grupos en Windows 2000 se explican en el Apartado 5.6.1.

El hecho de que cada archivo o carpeta tenga dos DACL en vez de una tiene que ver con el mecanismo de la *herencia de permisos* que incorpora Windows 2000: cada archivo o carpeta puede heredar implícitamente los permisos establecidos para la carpeta que lo contiene y puede además definir permisos propios (denominados explícitos en la jerga de Windows 2000). Es decir, que cada archivo o carpeta puede poseer potencialmente una *DACL heredada* y una *DACL explícita* (aunque no está obligado a ello, como veremos). De esta forma, si una cierta carpeta define permisos explícitos, éstos (junto con sus permisos heredados) serán a su vez los permisos heredados de sus subcarpetas y archivos (y así sucesivamente). El mecanismo de herencia de permisos es dinámico, queriendo decir que la modificación un permiso explícito de una carpeta se refleja en el correspondiente permiso heredado de sus subcarpetas y archivos.

La definición de permisos a archivos y carpetas sigue una serie de reglas:

- Cuando se crea un nuevo archivo o carpeta, este posee por defecto permisos heredados (de la carpeta o unidad donde se ubica) y ningún permiso explícito.
- Cualquier usuario que posea control total sobre el archivo o carpeta (por defecto, su propietario) puede incluir nuevos permisos (positivos o negativos) en la lista de permisos explícita.
- El control sobre la herencia de permisos (i.e., qué objetos heredan y qué permisos se heredan) se realiza a dos niveles:

- 1) Cada objeto (archivo o carpeta) tiene la potestad de decidir si desea o no heredar los permisos de su carpeta padre. Es decir, cada carpeta/archivo puede *desactivar* la herencia de su carpeta padre.
  - 2) Cuando se define un permiso explícito en una carpeta, se puede también decidir qué objetos por debajo van a heredarlo. En este caso, se puede decidir entre cualquier combinación de tres elementos: la propia carpeta, las subcarpetas y los archivos. La opción por defecto es todos, es decir, la carpeta y todas las subcarpetas y archivos.
- Copiar un archivo o carpeta a otra ubicación se considera una creación, y por tanto el archivo copiado recibe una lista de permisos explícitos vacía y se activa la herencia de la carpeta (o unidad) padre correspondiente a la nueva ubicación.
  - Mover un archivo distingue dos casos: si movemos una carpeta o archivo a otra ubicación dentro del mismo volumen (partición) NTFS, se desactiva la herencia y se mantienen los permisos que tuviera como explícitos en la nueva ubicación. Si el volumen destino es distinto, entonces se actúa como en una copia (sólo se tienen los permisos heredados de la carpeta padre correspondiente a la nueva ubicación).

### 5.6.1. Permisos Estándar y Permisos Individuales

Windows 2000 distingue entre los *permisos estándar* de carpetas (directorios) y los de archivos. Como ocurría en NT 4.0, los permisos estándar son combinaciones predefinidas de *permisos individuales*, que son permisos que controlan cada una de las acciones individuales que se pueden realizar sobre carpetas y archivos. La existencia de estas combinaciones predefinidas es el resultado de una agrupación “lógica” de los permisos individuales para facilitar la labor de administrar permisos.

En la Tabla 5.2 se muestran los permisos estándar de carpetas y archivos junto con su significado cualitativo. Las descripciones de las tablas hacen referencia a las acciones que cada permiso concede, pero no olvidemos que en Windows 2000 cada permiso puede ser positivo o negativo, es decir, que realmente cada permiso permite *o deniega* la acción correspondiente. Como puede verse en ambas tablas, muchos de los permisos estándar se definen de forma *incremental*, de forma que unos incluyen y ofrece un nivel de acceso superior que los anteriores. La herencia de permisos se establece de forma natural: las carpetas heredan directamente los permisos estándar establecidos en la carpeta padre, mientras que los archivos heredan cualquier permiso de su carpeta padre, excepto el de *Listar* (sólo definido para carpetas).

Cuando la asignación de permisos que queremos realizar no se ajusta al comportamiento de ninguno de los permisos estándar, debemos entonces ir directamente a asignar permisos individuales. La Tabla 5.3 muestra cuáles son los permisos individuales en Windows 2000, junto con su significado concreto. También en este caso debe entenderse que cada permiso puede ser concedido de forma positiva o negativa.

CARPETAS	
Nombre	Significado
Listar	Permite listar la carpeta: ver los archivos y subcarpetas que contiene.
Leer	Permite ver el contenido de los archivos y subcarpetas, así como su propietario, permisos y atributos (sistema, sólo lectura, oculto, etc.).
Escribir	Permite crear nuevos archivos y subcarpetas. Permite modificar los atributos de la propia carpeta, así como ver su propietario, permisos y atributos.
Leer y Ejecutar	Permite moverse por la jerarquía de subcarpetas a partir de la carpeta, incluso si no se tienen permisos sobre ellas. Además, incluye todos los permisos de <i>Leer</i> y de <i>Listar</i> .
Modificar	Permite eliminar la carpeta más todos los permisos de <i>Escribir</i> y de <i>Leer y Ejecutar</i> .
Control Total	Permite cambiar permisos, tomar posesión y eliminar subcarpetas y archivos (aun no teniendo permisos sobre ellos), así como todos los permisos anteriores.

ARCHIVOS	
Nombre	Significado
Leer	Permite ver el contenido del archivo, así como su propietario, permisos y atributos (sistema, sólo lectura, oculto, etc.).
Escribir	Permite sobrescribir el archivo, modificar sus atributos y ver su propietario, permisos y atributos.
Leer y Ejecutar	Permite ejecutar el archivo más todos los permisos de <i>Leer</i> .
Modificar	Permite modificar y eliminar el archivo más todos los permisos de <i>Escribir</i> y de <i>Leer y Ejecutar</i> .
Control Total	Permite cambiar permisos, tomar posesión, más todos los permisos anteriores.

**Cuadro 5.2:** Permisos estándar sobre carpetas y archivos en Windows 2000

Por último, la Tabla 5.4 pone de manifiesto el subconjunto de los permisos individuales forman cada uno de los permisos estándar mencionados anteriormente. Como curiosidad, puede verse que los permisos individuales correspondientes a *Listar* y *Leer y Ejecutar* son los mismos. En realidad, lo que les distingue es cómo se heredan: el primero sólo es heredado por carpetas, mientras que el segundo es heredado por carpetas y archivos.

## 5.7. Reglas de Protección

Las principales reglas que controlan la comprobación de permisos a carpetas y archivos son las siguientes:

- Una única acción de un proceso puede involucrar varias acciones individuales sobre varios archivos y/o carpetas. En ese caso, el sistema verifica si el proce-

Nombre	Significado
Atravesar carpeta/Ejecutar archivo	Aplicado a una carpeta, permite moverse por subcarpetas en las que puede que no se tenga permiso de acceso. Aplicado a un archivo, permite su ejecución.
Leer carpeta/Leer datos	Aplicado a una carpeta, permite ver los nombres de sus ficheros y subcarpetas. Aplicado a un archivo, permite leer su contenido.
Leer atributos	Permite ver los atributos del fichero/carpeta, tales como <i>oculto</i> o <i>sólo lectura</i> , definidos por NTFS.
Leer atributos extendidos	Permite ver los atributos extendidos del archivo o carpeta. (Estos atributos están definidos por los programas y pueden variar).
Crear ficheros/Escribir datos	Aplicado a una carpeta, permite crear archivo en ella. Aplicado a un archivo, permite modificar y sobrescribir su contenido.
Crear carpetas/Anexar datos	Aplicado a una carpeta, permite crear subcarpetas en ella. Aplicado a un archivo, permite añadir datos al final del mismo.
Escribir atributos	Permite modificar los atributos de un archivo o carpeta.
Escribir atributos extendidos	Permite modificar los atributos extendidos de un archivo o carpeta.
Borrar subcarpetas y archivos	Sólo se puede aplicar a una carpeta, y permite borrar archivos o subcarpetas de la misma, aun no teniendo permiso de borrado en dichos objetos.
Borrar	Permite eliminar la carpeta o archivo.
Leer permisos	Permite leer los permisos de la carpeta o archivo.
Cambiar permisos	Permite modificar los permisos de la carpeta o archivo.
Tomar posesión	Permite tomar posesión de la carpeta o archivo.

**Cuadro 5.3:** Permisos individuales en Windows 2000

so tiene o no permisos para todas ellas. Si le falta algún permiso, la acción se rechaza con un mensaje de error genérico de falta de permisos.

- Los permisos en Windows 2000 son acumulativos: un proceso de usuario posee implícitamente todos los permisos correspondientes a los SIDs de su acreditación (ver Apartado 5.4), es decir, los permisos del usuario y de todos los grupos a los que pertenece.
- La ausencia un cierto permiso sobre un objeto supone implícitamente la imposibilidad de realizar la acción correspondiente sobre el objeto.
- Si se produce un conflicto en la comprobación de los permisos, los permisos negativos tienen prioridad sobre los positivos, y los permisos explícitos tienen prioridad sobre los heredados.

Estas reglas son más fáciles de recordar si se conoce el algoritmo que sigue Windows 2000 para conceder o denegar una acción concreta sobre un archivo o directorio



Permiso	C.Total	Modif.	L/Ej.	Listar	Leer	Escribir
Atravesar carpeta/ejecutar archivo	✓	✓	✓	✓		
Leer carpeta/Leer datos	✓	✓	✓	✓	✓	
Leer atributos	✓	✓	✓	✓	✓	
Leer atributos extendidos	✓	✓	✓	✓	✓	
Crear ficheros/escribir datos	✓	✓				✓
Crear carpetas/anexar datos	✓	✓				✓
Escribir atributos	✓	✓				✓
Escribir atributos extendidos	✓	✓				✓
Borrar subcarpetas y archivos	✓					
Borrar	✓	✓				
Leer permisos	✓	✓	✓	✓	✓	✓
Cambiar permisos	✓					
Tomar posesión	✓					

**Cuadro 5.4:** Correspondencia entre permisos estándar e individuales en Windows 2000.

concreto. Para ello, el sistema explora secuencialmente las entradas de las DACLs de dicho objeto hasta que se cumple alguna de las condiciones siguientes:

1. Cada permiso involucrado en la acción solicitada está concedido explícitamente al SID del usuario o de algún grupo al que el usuario pertenece. En ese caso, se permite la acción.
2. Alguno de los permisos involucrados está explícitamente denegado para el SID del usuario o para alguno de sus grupos. En este caso, se deniega la acción.
3. La lista (DACL) ha sido explorada completamente y no se ha encontrado una entrada (ni positiva ni negativa) correspondiente a alguno de los permisos involucrados en la acción para el SID del usuario o sus grupos. En este caso, se deniega la acción.

Este algoritmo realmente produce el comportamiento descrito por las reglas anteriores debido al orden en que Windows 2000 establece las entradas de las DACLs de cada objeto. Este orden es siempre el siguiente: permisos negativos explícitos, permisos positivos explícitos, permisos negativos heredados y permisos positivos heredados.



# Capítulo 6

## Montaje de Dispositivos en Linux

### Índice General

---

<b>6.1. Introducción . . . . .</b>	<b>57</b>
<b>6.2. Utilización Básica . . . . .</b>	<b>57</b>
<b>6.3. La tabla de Montaje de Dispositivos . . . . .</b>	<b>58</b>
<b>6.4. Otros mandatos relacionados . . . . .</b>	<b>60</b>

---



## 6.1. Introducción

Unix ofrece al usuario una visión jerárquica del sistema de archivos, donde los directorios pueden ser creados en cualquier nivel para organizar así los ficheros. A diferencia de otros sistemas, Unix ofrece una sola jerarquía, es decir, un solo directorio raíz.

Por contra, en sistemas tipo Windows, el sistema presenta varias jerarquías, tantas como unidades de almacenamiento (particiones, disquetes, cd-roms, unidades de red) tenga el ordenador. Cada una de estas jerarquías es identificada por una letra de unidad seguida del carácter ':', por ejemplo, D:.

Unix, en cambio, encadena todos los dispositivos que contienen sistemas de archivos en una única jerarquía, siendo por tanto este acceso transparente al usuario. A esta técnica se le denomina *montaje de dispositivos*.

## 6.2. Utilización Básica

Para utilizar un dispositivo que contiene un sistema de archivos se utiliza el mandato `mount`, el cual utiliza dos argumentos, un nombre de dispositivo y un nombre de directorio. Por ejemplo:

```
mount /dev/hdb6 /mnt
```

En el ejemplo anterior, el directorio raíz del sistema de archivos que reside en el dispositivo `/dev/hdb6` (la segunda partición lógica del disco duro primario esclavo) se monta en el directorio `/mnt`. De esta forma, cualquier acceso que indique `/mnt` hace que Unix acceda a la partición que ha sido montada.

Linux reconoce muchos tipos de sistemas de archivos, algunos nativos del propio Linux y otros propios de otros sistemas operativos o estándares internacionales. Los más relevantes son:

Tipo	Descripción
ext2	Sistema de archivos nativo Linux.
ext3	ext2 con registro de transacciones
swap	Área de intercambio en disco de Linux.
proc	Jerarquía de información sobre el núcleo.
msdos	Sistema FAT 16 o FAT 32 con nombres cortos.
vfat	Sistema FAT 16 o FAT 32 con nombres largos.
iso9660	Sistema de archivos en CD-ROM.
nfs	Directorio remoto accesible vía NFS.

**Cuadro 6.1:** Principales tipos de particiones utilizables desde Linux.

Para indicar a `mount` el tipo del sistema de archivos, se utiliza la opción `-t`. Por ejemplo:

```
mount -t iso9660 /dev/cdrom1 /mnt/cdrom
```

Al margen de los tipos anteriores puede utilizarse el tipo `auto`, el cual indica a `mount` que reconozca de forma automática el tipo del sistema de archivos, opción que resulta especialmente útil en el fichero `/etc/fstab`, el cual se describe a continuación. Cuando se utiliza `mount` sin la opción `-t`, se asume la opción `-t auto`.

Debe tenerse en cuenta que lo anteriormente expuesto incluye a todo tipo de dispositivos. Por tanto, el acceso a medios no fijos como cd-rom o disquete se realiza igualmente utilizando `mount`.

Una utilidad adicional `mount` consiste en visualizar la tabla de los dispositivos actualmente montados. Para ello, basta con ejecutar `mount` sin argumentos.

La acción contraria a montar un dispositivo se llama, lógicamente, desmontar el dispositivo. Para ello se invoca el mandato `umount` indicando el directorio donde el dispositivo había sido montado. Siguiendo con el ejemplo anterior, antes de poder expulsar el cd-rom, deberíamos ejecutar lo siguiente:

```
umount /mnt/cdrom
```

### 6.3. La tabla de Montaje de Dispositivos

Con el fin de simplificar y unificar la gestión de los dispositivos montados, el administrador puede establecer en el fichero `/etc/fstab` aquellos dispositivos que son conocidos por el sistema, llegando incluso a indicar si algunos de ellos deben ser montados durante el momento del arranque del ordenador. Un ejemplo de este fichero es el siguiente:

```

/dev/hda3      /          ext2    defaults    1 1
/dev/hda2      swap        swap     defaults    0 0
none          /proc       proc     defaults    0 0
/dev/hdb1      /e7001     ext2     noauto,user  0 0
/dev/hdb2      /e700      msdos    noauto,user  0 0
/dev/fd0       /A         auto     noauto,user  0 0

```

Cada línea de dicho fichero establece un dispositivo a montar junto con sus opciones. Por ejemplo, la Tabla 6.2 interpreta el significado de los parámetros de la primera línea.

Parámetro	Descripción
/dev/hda3	Dispositivo a montar.
/	Directorio donde montarlo.
ext2	Tipo del sistema de archivos.
defaults	Opciones de montaje.
1	Volcado.
1	Orden de verificación por <code>fsck</code> en el arranque del sistema. Sólo debe aplicarse a los dispositivos locales que se montan durante el arranque.

**Cuadro 6.2:** Interpretación del fichero `/etc/fstab`.

Existe una gran cantidad de opciones de montaje, algunas de las cuales son dependientes del tipo del sistema de archivos (consúltase el mandato `mount` en el manual). Las opciones más comunes y su significado se presentan en la Tabla 6.3.

Opción	Descripción
defaults	Opciones por defecto. Incluye montar el dispositivo durante el arranque del ordenador
auto/noauto	Si/No montar el dispositivo durante el arranque
user/nouser	Si/No permitir a un usuario convencional montar este dispositivo
ro	Montar el dispositivo en modo de solo-lectura
rw	Montar el dispositivo en modo de lectura-escritura
dev/nodev	Si/No interpretar ficheros especiales de dispositivo
suid/nosuid	Si/No permitir la ejecución de ficheros con bits SETUID o SETGID activos

**Cuadro 6.3:** Opciones más comunes en el montaje de dispositivos en Linux.

Una de las ventajas de este fichero es que simplifica la utilización del mandato `mount`, el cual puede utilizar un solo argumento, bien sea el nombre del dispositivo

o, mejor aún, el nombre del directorio. En estos casos, `mount` utiliza el tipo de sistema de archivos y las opciones especificados en el fichero. Por ejemplo:

```
mount /A
```

monta en el directorio `/A` el dispositivo `/dev/fd0` de acuerdo con la última línea del fichero `/etc/fstab` anterior. Puesto que se ha utilizado el tipo de sistema de archivos `auto`, el disquete podría contener indistintamente un sistema de archivos `msdos`, `ext2`, `vfat`, etc., siendo la detección del tipo realizada por `mount`.

## 6.4. Otros mandatos relacionados

- `mkfs`: Crea un sistema de archivos vacío en un dispositivo.
- `fsck`: Verifica la consistencia de un sistema de archivos.
- `df`: Informa sobre el espacio libre y ocupado de un sistema de archivos.



# Capítulo 7

## Dominios en Linux

### Índice General

---

<b>7.1. Concepto de Dominio</b>	<b>61</b>
<b>7.2. Dominios en Linux con OpenLDAP</b>	<b>62</b>
7.2.1. Introducción	62
7.2.2. Servicios de Directorio y LDAP	62
7.2.3. Configuración Básica de OpenLDAP	65
7.2.4. Implementación de un Dominio Linux con OpenLDAP	73
7.2.5. Herramientas Gráficas de Administración	75
<b>7.3. Network File System (NFS)</b>	<b>76</b>
7.3.1. Cómo Funciona NFS	77
7.3.2. Instalación y Configuración del Cliente NFS	77
7.3.3. Instalación y Configuración del Servidor NFS	78

---

### 7.1. Concepto de Dominio

En el mundo Linux no existe un concepto de dominio tan elaborado como en el mundo de Windows 2000. Sin embargo, se consigue un efecto similar al activar un servicio en una máquina Linux (que actuaría como “servidor” de cuentas y grupos) y otro

servicio que permite la exportación de directorios a máquinas remotas. En concreto, dichos servicios se denominan LDAP y NFS, respectivamente. Ambos son explicados a continuación.

## 7.2. Dominios en Linux con OpenLDAP

### 7.2.1. Introducción

Desde el punto de vista de la administración de sistemas, suele denominarse *dominio* a un conjunto de equipos interconectados que comparten información administrativa (usuarios, grupos, contraseñas, etc.) centralizada. Ello requiere fundamentalmente la disponibilidad de (al menos) un ordenador que almacene físicamente dicha información y que la comunique al resto cuando sea necesario, típicamente mediante un esquema cliente-servidor. Por ejemplo, cuando un usuario desea iniciar una conexión interactiva en cualquiera de los ordenadores (clientes) del dominio, dicho ordenador deberá validar las credenciales del usuario en el servidor, y obtener de éste todos los datos necesarios para poder crear el contexto inicial de trabajo para el usuario.

En Windows 2000, la implementación del concepto de dominio se realiza mediante el denominado *Directorio Activo*, un servicio de directorio basado en diferentes estándares como LDAP (Lightweight Directory Access Protocol) y DNS (Domain Name System). En el mundo Unix, los dominios solían implementarse mediante el famoso *Network Information System* (NIS), del que existían múltiples variantes. Sin embargo, la integración de servicios de directorio en Unix ha posibilitado la incorporación de esta tecnología, mucho más potente y escalable que NIS, en la implementación de dominios.

Este capítulo describe cómo una implementación libre del protocolo LDAP para Unix, denominada *OpenLDAP* ([www.openldap.org](http://www.openldap.org)), puede utilizarse para implementar dominios en RedHat Linux.

### 7.2.2. Servicios de Directorio y LDAP

En el contexto de las redes de ordenadores, se denomina *directorio* a una base de datos especializada que almacena información sobre los recursos, u “objetos”, presentes en la red (tales como usuarios, ordenadores, impresoras, etc.) y que pone dicha información a disposición de los usuarios de la red. Por este motivo, esta base de datos suele estar optimizada para operaciones de búsqueda, filtrado y lectura más que para operaciones de inserción o transacciones complejas. Existen diferentes estándares que especifican servicios de directorio, siendo el denominado *X.500* tal vez el más conocido.

El estándar X.500 define de forma nativa un protocolo de acceso denominado DAP (Directory Access Protocol) que resulta muy complejo (y computacionalmente pesado) porque está definido sobre la pila completa de niveles OSI. Como alternativa a

DAP para acceder a directorios de tipo X.500, LDAP (Lightweight Directory Access Protocol) ofrece un protocolo “ligero” casi equivalente, pero mucho más sencillo y eficiente, diseñado para operar directamente sobre TCP/IP. Actualmente, la mayoría de servidores de directorio X.500 incorporan LDAP como uno de sus protocolos de acceso.

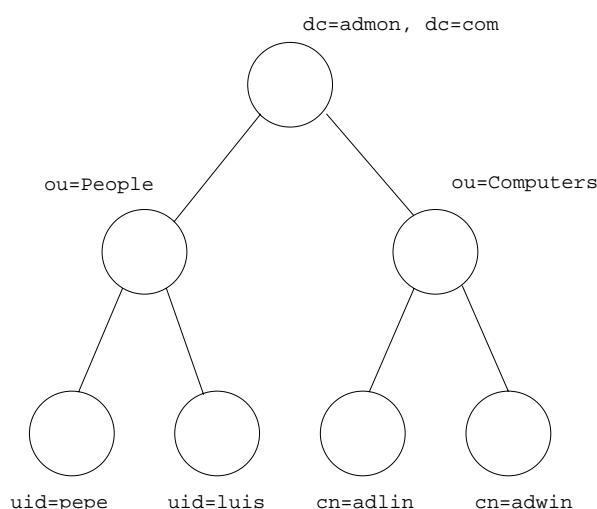
LDAP permite el acceso a la información del directorio mediante un esquema cliente-servidor, donde uno o varios servidores mantienen la misma información de directorio (actualizada mediante réplicas) y los clientes realizan consultas a cualquiera de ellos. Ante una consulta concreta de un cliente, el servidor contesta con la información solicitada y/o con un “puntero” donde conseguir dicha información o datos adicionales (normalmente, el “puntero” es otro servidor de directorio).

Internamente, el modelo de datos de LDAP (derivado de X.500, pero algo restringido) define una estructura jerárquica de objetos o *entradas* en forma de árbol, donde cada objeto o entrada posee un conjunto de atributos. Cada atributo viene identificado mediante un *nombre* o acrónimo significativo, pertenece a un cierto *tipo* y puede tener uno o varios *valores* asociados. Toda entrada viene identificada unívocamente en la base de datos del directorio mediante un atributo especial denominado *nombre distinguido* o *dn* (distinguished name). El resto de atributos de la entrada depende de qué objeto esté describiendo dicha entrada. Por ejemplo, las entradas que describen personas suelen tener, entre otros, atributos como *cn* (common name) para describir su nombre común, *sn* (surname) para su apellido, *mail* para su dirección de correo electrónico, etc. La definición de los posibles tipos de objetos, así como de sus atributos (incluyendo su nombre, tipo, valor(es) admitido(s) y restricciones), que pueden ser utilizados por el directorio de un servidor de LDAP la realiza el propio servidor mediante el denominado *esquema* del directorio. Es decir, el esquema contiene las definiciones de los objetos que pueden darse de alta en el directorio.

El nombre distinguido de cada entrada del directorio es una cadena de caracteres formada por pares `<tipo_atributo>=<valor>` separados por comas, que representa la ruta invertida que lleva desde la posición lógica de la entrada en el árbol hasta la raíz del mismo. Puesto que se supone que un directorio almacena información sobre los objetos que existen en una cierta organización, cada directorio posee como raíz (o *base*, en terminología LDAP) la ubicación de dicha organización, de forma que la base se convierte de forma natural en el *sufijo* de los nombres distinguidos de todas las entradas que mantiene el directorio. Existen dos formas de nombrar, o estructurar, la raíz de un directorio LDAP:

- a) Nombrado “tradicional”: formado por el país y estado donde se ubica la organización, seguida por el nombre de dicha organización. Por ejemplo, la raíz o base de la Universidad Politécnica de Valencia podría ser algo así: “o=UPV, st=Valencia, c=ES”.
- b) Nombrado basado en nombres de dominio de Internet (es decir, en DNS): este nombrado utiliza los dominios DNS para nombrar la raíz de la organización. En este caso, la base de la UPV sería: “dc=upv, dc=es”. Este es el nombrado que vamos a utilizar puesto que permite localizar a los servidores de LDAP utilizando búsquedas DNS.

A partir de esa base, el árbol se subdivide en los nodos y subnodos que se estime oportuno para estructurar de forma adecuada los objetos de la organización, objetos que se ubican finalmente como las hojas del árbol. De esta forma, el nombre distinguido de cada entrada describe su posición en el árbol de la organización (y vice-versa), de forma análoga a un sistema de archivos típico, en el que el nombre absoluto (único) de cada archivo equivale a su posición en la jerarquía de directorios del sistema, y vice-versa. En la Figura 7.1 se muestra un ejemplo de un directorio sencillo (dos usuarios y dos equipos) de la organización `admon.com`.



**Figura 7.1:** Estructura de directorio del dominio `admon.com`.

De acuerdo con dicha figura, la entrada correspondiente al usuario “pepe” tendría como nombre distinguido “uid=pepe, ou=People, dc=admon, dc=com”. Al margen de ese identificador único, cada entrada u objeto en el directorio puede tener, como hemos dicho, un conjunto de atributos tan descriptivo como se desee. Cada objeto necesita, al margen de su nombre distinguido, su “clase de objeto”, que se especifica mediante el atributo `ObjectClass`<sup>1</sup>. Este atributo especifica implícitamente el resto de atributos de dicho objeto, de acuerdo con la definición establecida en el esquema. Siguiendo con el ejemplo anterior, a continuación se muestra un subconjunto de los

---

<sup>1</sup>Un mismo objeto puede pertenecer a diferentes clases simultáneamente, por lo que pueden existir múltiples atributos de este tipo para un mismo objeto en el directorio.

atributos del usuario “pepe”:

```
dn: uid=pepe, ou=People, dc=admon, dc=com
objectClass: person
cn: Jose García
sn: García
description: alumno
mail: pepe@admon.com
```

El formato en el que se han mostrado los atributos del objeto se denomina LDIF (LDAP Data Interchange Format), y resulta útil conocerlo porque es el formato que los servidores LDAP (y OpenLDAP entre ellos) utilizan por defecto para insertar y extraer información del directorio.

Puesto que nosotros vamos a utilizar el directorio con un uso muy específico (centralizar la información administrativa de nuestro dominio para autenticar usuarios de forma global) deberíamos asegurarnos que en el esquema de nuestro directorio existen los tipos de objetos (y atributos) adecuados para ello. Afortunadamente, OpenLDAP posee por defecto un esquema suficientemente rico para cubrir este cometido. Por ejemplo, cada usuario puede definirse mediante un objeto de tipo `posixAccount`, que posee entre otros atributos para almacenar su UID, GID, contraseña, etc. A título de curiosidad, la entrada en el esquema que define el atributo para la contraseña (`userPassword`) se muestra a continuación:

```
attributetype (2.5.5.35 NAME 'userPassword'
    EQUALITY octetStringMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.40{128})
```

### 7.2.3. Configuración Básica de OpenLDAP

La utilización de LDAP para centralizar las labores administrativas (es decir, la creación de un “dominio Linux”) tiene dos partes bien diferenciadas: primero hay que instalar y configurar uno o varios ordenadores del futuro dominio como *servidores de LDAP* y el resto de ordenadores del dominio como *clientes* de dicha información. Una vez conseguido este objetivo, el segundo paso es configurar los clientes para que utilicen a los servidores como fuente de información administrativa (cuentas de usuarios y grupos, contraseñas, hosts, etc.) en vez de (o mejor dicho, además de) sus ficheros de configuración locales.

En ambos casos, es importante resaltar que parte de la configuración que se describe a continuación es *dependiente* del software de LDAP concreto que se va a utilizar (en este caso, OpenLDAP) y de la versión de UNIX utilizada (RedHat Linux). La configuración de otros servidores de LDAP, en la misma u otras versiones de UNIX, puede ser distinta a la que aquí se va a exponer.

## Configuración de OpenLDAP con Servidor Unico

En el caso más sencillo, nuestra red puede requerir un solo servidor de LDAP, siendo el resto de los ordenadores clientes del mismo. Si este es el caso, esta sección expone los pasos a seguir para configurar el servidor y los clientes.

### Primer Paso: Instalación del Servidor

El paquete que incorpora el servidor de OpenLDAP se denomina `openldap-servers` y puede encontrarse en la distribución correspondiente de RedHat Linux.

Una vez descargado e instalado, este paquete instala los ficheros de configuración por defecto bajo el directorio `/etc/openldap`, crea un directorio vacío denominado `/lib/var/ldap` para albergar la base de datos con la información del directorio y sus índices, y finalmente incorpora el servicio o “demonio” de LDAP, denominado `slapd`. Curiosamente, el nombre del servicio presente en `/etc/rc.d/init.d`, y que utilizaremos para iniciar y parar el demonio, se denomina `ldap` y no `slapd`.

### Segundo Paso: Configuración del Servidor

La configuración del servicio `slapd` se realiza en `/etc/openldap/slapd.conf` fundamentalmente. Este fichero contiene referencias a los demás ficheros necesarios para el servicio `slapd` (como por ejemplo, las definiciones del esquema), y un conjunto de secciones donde se describen los directorios de los que se mantiene información. Es incluso posible almacenar los directorios en bases de datos de distintos formatos internos y definir opciones específicas diferentes para cada una. En el caso más sencillo, sin embargo, tendremos un único directorio almacenado en una base de datos en el formato por defecto (ldb), cuyas opciones no modificaremos.

De hecho, de los numerosos parámetros que pueden configurarse en este fichero, sólo vamos a comentar los estrictamente necesarios para configurar un servidor básico de LDAP que luego haga de servidor de un dominio Linux. Para configurar cualquier otro parámetro avanzado, se recomienda la lectura previa del documento “OpenLDAP Administrator’s Guide”, que puede encontrarse en [www.openldap.org](http://www.openldap.org).

En definitiva, los cinco parámetros fundamentales que es necesario configurar son los siguientes (el resto pueden dejarse con sus opciones por defecto):

- 1) **Sufijo.** Este es el sufijo de las entradas del directorio, es decir, lo que hemos denominado base o raíz del directorio. Por ejemplo, para el dominio “admon.com” deberíamos configurar:

```
suffix    "dc=admon, dc=com"
```

- 2) **Directorio de la(s) base(s) de datos.** Se especifica mediante la directiva siguiente:

```
directory /var/lib/ldap
```

- 3) **Cuenta del administrador** (del directorio). Esta es la cuenta del usuario administrador del directorio, lógicamente en formato de LDAP. Las credenciales que se sitúan en esta opción (y la siguiente) tienen validez independientemente de

que este usuario exista realmente en la base de datos del directorio o no. El nombre por defecto es “manager”, pero si queremos lo podemos cambiar por “root” (a la UNIX):

```
rootdn "cn=root, dc=admon, dc=com"
```

Sin embargo, no hay que confundir a este usuario con el usuario “root” del sistema Linux donde estamos instalando el servidor de LDAP. Más adelante, si queremos, podemos hacerlos coincidir.

- 4) **Contraseña del administrador.** Cuando las operaciones a realizar en la base de datos no permitan una conexión anónima (sin acreditarse), y en concreto, cuando necesiten del usuario “administrador del directorio” definido arriba, dichas operaciones deben acompañarse de la contraseña de esta cuenta, que se especifica en la siguiente opción:

```
rootpw <CONTRASEÑA>
```

Hay que tener en cuenta que la contraseña que pongamos aquí será visible por cualquier usuario que pueda leer el fichero (aunque por defecto, éste es un permiso sólo de root). Por tanto, es conveniente ponerla cifrada. Para ello, si queremos utilizar la misma contraseña que tiene “root”, podemos por ejemplo copiar su contraseña cifrada del archivo `/etc/shadow` y pegarla así:

```
rootpw {crypt}<CONTRASEÑA_CIFRADA>
```

O, aún mejor, podemos crear una contraseña nueva mediante la orden:

```
slappasswd -h {MD5}
```

Que nos pide una contraseña y nos la muestra cifrada. Luego simplemente copiamos el resultado de dicha orden tras la cadena, cambiando en este caso “crypt” por “MD5” (no debe haber espacios en blanco entre la cadena que indica el tipo de cifrado y la contraseña).

- 5) **Niveles de acceso.** Esta opción permite especificar, con un grano muy fino, a quién se da el permiso para leer, buscar, comparar, modificar, etc., la información almacenada en el directorio. Para ello se utilizan una o múltiples reglas de control de acceso, cuya sintaxis se muestra a continuación:

```
access to <what> [by <who> <access_control>]+
```

donde:

- `<what>` es una expresión que especifica a qué datos del directorio se aplica la regla. Existen tres opciones: (1) puede indicarse todo el directorio mediante un asterisco (\*), (2) un subconjunto de entradas cuyo nombre distinguido contiene un cierto sufijo (por ejemplo, `dn=".*, dc=admon, dc=com"`) o (3) un atributo concreto de dichos objetos (por ejemplo, `dn=".*, ou=People, dc=admon, dc=com" attr=userPassword`).

- `<who>` indica a quién (a qué *usuario(s)*) se especifica la regla. Puede tomar diferentes valores, siendo los más comunes los siguientes: `self` (el propietario de la entrada), `dn="..."` (el usuario representado por el nombre distinguido), `users` (cualquier usuario acreditado), `anonymous` (cualquier usuarios no acreditado) y `*` (cualquier usuario).
- `<access_control>` indica qué operación concede la regla: `none` (sin acceso), `auth` (utilizar la entrada para validarse), `compare` (comparar), `search` (búsqueda), `read` (lectura), y `write` (modificación).

Para entender correctamente la semántica de las reglas de control de acceso, es imprescindible conocer el método que sigue `slapd` para evaluarlas, cada vez que recibe una petición por parte de un cliente: primero se busca, secuencialmente, la primera regla cuya expresión `<what>` incluye la entrada, o entradas, afectadas por la petición. Dentro de ella, se busca secuencialmente la primera expresión `<who>` que incluye al *usuario* que ha realizado la petición desde el cliente. Una vez encontrado, si el nivel de acceso expresado por `<access_control>` es mayor o igual al requerido por la petición, entonces ésta se concede, y en caso contrario se deniega. Si no existe ninguna expresión `<who>` que incluya al usuario, o bien no existe ninguna regla cuya expresión `<what>` incluya la información afectada por la petición, se deniega la petición.

Por tanto, el orden en que aparecen las reglas en el fichero, y el orden interno de las cláusulas “by” dentro de cada regla, es relevante: si varias reglas afectan a los mismos objetos, las reglas más específicas deberían ubicarse antes en el fichero; y, dentro de una regla, si incluimos varias cláusulas by, también debemos situar las más específicas primero.

Un ejemplo razonable de reglas de acceso podría ser el siguiente:

```
access to dn=".*,ou=People,dc=admon,dc=com" attr=userPassword
    by self write
    by dn="cn=root,dc=admon,dc=com" write
    by * auth

access to dn=".*,ou=People,dc=admon,dc=com"
    by dn="cn=root,dc=admon,dc=com" write
    by * read

access to *
    by dn="cn=root,dc=admon,dc=com" write
    by * read
```

En este ejemplo, la primera regla de acceso permite a cada usuario a cambiarse su propia contraseña<sup>2</sup>, al administrador cambiar la de cualquier usuario y al resto de usuarios sólo pueden utilizar este campo para autenticarse. La segunda regla de acceso se aplica al resto de los atributos de las cuentas de usuario, y permite al administrador su cambio y al resto sólo su consulta (de esta forma

<sup>2</sup>La contraseña es el atributo `userPassword` en los objetos de tipo usuario, que se sitúan por defecto dentro de la unidad organizativa “People”.



evitamos que un usuario pueda cambiarse su UID, GID, etc.). Finalmente, la tercera regla permite al administrador cambiar cualquier entrada del directorio y al resto de usuarios sólo leerlas.

### Tercer Paso: Comprobación de la Configuración

Una vez realizada la configuración del apartado anterior, ya estamos en condiciones de iniciar el servicio `slapd` y comprobar que, de momento, todo funciona correctamente. Para ello:

```
service ldap start
```

Si el servicio ha arrancado correctamente, y a pesar de que actualmente el directorio está vacío, deberíamos ser capaces de preguntar al menos por un tipo de atributo denominado “namingContexts”. Para ello, utilizamos la orden `ldapsearch`:

```
ldapsearch -x -b '' -s base '(objectclass=*)' namingContexts
```

Si todo funciona bien, el resultado debería ser algo así:

```
#
# filter: (objectclass=*)
# requesting: namingContexts
#
#
# dn: namingContexts: dc=admon, dc=com
# search result
search: 2
result: 0 success
# numResponses: 2
# numEntries: 1
```

Es imprescindible que en la salida por pantalla aparezca en sufijo del dominio (`dc=admon,dc=com` en el ejemplo). Si no es así, lo más probable es que las reglas de control de acceso especificadas en el fichero de configuración de `slapd` no admitan la lectura del directorio a usuarios no autenticados (en la orden anterior estamos accediendo sin credenciales conocidas, es decir, de forma anónima). Para acceder con credenciales concretas, se utilizan las opciones `-D` y `-W`, como por ejemplo:

```
-D "cn=root,dc=admon,dc=com" -W
```

que utiliza el `dn` especificado y nos pide su contraseña de forma interactiva.

Una vez el servicio funciona correctamente, debemos iniciarlo por defecto en los niveles de ejecución 3 y 5. Para ello puede utilizarse la herramienta gráfica `serviceconf` o, en línea de órdenes:

```
chkconfig --level 35 ldap on
```

### Cuarto Paso: Configuración de los Clientes

La configuración de los clientes de OpenLDAP se realiza alternativamente en dos ficheros de configuración: `/etc/openldap/ldap.conf` y `/etc/ldap.conf`. El primero sirve de fichero de configuración por defecto para todas las herramientas clientes de OpenLDAP, mientras que el segundo incluye opciones de configuración más específicas para autenticación, gestión de contraseñas, conexiones cifradas, etc.

En principio, de momento sólo necesitaríamos configurar un par de opciones del primero de ambos ficheros en todos los ordenadores clientes, incluyendo el servidor o servidores de LDAP:

```
BASE dc=admon,dc=com
HOST adlin.admon.com
```

Es importante hacer notar que OpenLDAP aún no soporta la localización del servidor basada en registros de servicio de DNS (como es el caso del Directorio Activo de Windows 2000). Por tanto, en la opción `HOST` es necesario especificar o bien una dirección IP o bien un nombre de ordenador que pueda resolverse correctamente sin utilizar el propio directorio (por ejemplo, que esté dado de alta en el fichero `/etc/hosts` del ordenador cliente, o que pueda resolverse correctamente mediante una consulta DNS).

### Quinto Paso: Migración de la Información del Servidor

El siguiente y último paso consiste en añadir al directorio, que aún está vacío, toda la información presente en el ordenador que está haciendo de servidor. Esto incluye todos los recursos dados de alta en sus ficheros de configuración, tales como cuentas de usuarios, grupos, ordenadores, etc., así como diferentes “contenedores” o “unidades organizativas” (OrganizationalUnits) que distribuyen los objetos de forma racional.

Para ello, OpenLDAP incorpora un conjunto de herramientas que pueden utilizarse para realizar esta migración de forma automática. El paso previo para ello es editar el fichero `/usr/share/openldap/migration/migrate_common.ph`. En concreto, tenemos que editar las dos directivas siguientes:

```
$DEFAULT_MAIL_DOMAIN = "admon.com";
$DEFAULT_BASE = "dc=admon,dc=com";
```

Una vez modificadas ambas, tenemos diferentes opciones para incorporar la información del sistema al directorio. Entre ellas, la más recomendable es realizar la migración por partes, añadiendo primero la “base” (es decir, las entradas correspondientes a la organización y sus unidades organizativas por defecto) y posteriormente migrando los usuarios, grupos, hosts, etc., que se ubicarán dentro de dichas unidades.

Para todo ello existen *scripts* de Perl en el directorio mencionado arriba (donde se ubica `migrate_common.ph`), que podemos utilizar de la siguiente forma (en todo el proceso, el servicio `ldap` debe estar ejecutándose):

- Para la migración de la base, se exporta primero sus objetos a un fichero, en

formato LDIF, y luego se insertan en el directorio mediante la orden `ldapadd`:

```
bash# ./migrate_base.pl > /root/base.ldif
bash# ldapadd -x -c -D "cn=root,dc=admon,dc=com"
-W -f /root/base.ldif
```

Nótese que con la opción `-D` estamos acreditándonos, para la operación, como el usuario “administrador del directorio”, y con la opción `-W` le decimos a la orden que nos pida la contraseña de dicho usuario de forma interactiva. La opción `-c` consigue que `ldapadd` siga insertando registros a pesar de que se produzcan errores en alguna inserción (cosa que suele ocurrir con el primer registro).

- Para la migración de los usuarios:

```
bash# ./migrate_passwd.pl /etc/passwd > /root/usuarios.ldif
```

Una vez copiados todos los usuarios en format LDIF, tenemos que editar el fichero `usuarios.ldif` y eliminar todos los registros que hacen referencia a usuarios especiales de Linux, incluyendo a “root” (no se recomienda en general exportar la cuenta de “root” mediante LDAP, por cuestiones de seguridad). En definitiva, sólo deberían quedar en el fichero los registros asociados a los usuarios comunes que hemos añadido al sistema. Una vez editado el fichero, añadimos los registros al directorio:

```
bash# ldapadd -x -c -D "cn=root,dc=admon,dc=com"
-W -f /root/usuarios.ldif
```

- Para la migración de los grupos:

```
bash# ./migrate_group.pl /etc/group > /root/grupos.ldif
```

Como con los usuarios, eliminamos del fichero `grupos.ldif` los grupos especiales, y dejamos sólo los grupos privados de los usuarios comunes que hemos añadido al sistema. Tras la modificación, añadimos esos grupos al directorio:

```
bash# ldapadd -x -c -D "cn=root,dc=admon,dc=com"
-W -f /root/grupos.ldif
```

- Y así sucesivamente con hosts, servicios, etc. De todas formas, para nuestros propósitos de uso del directorio, con la migración hecha hasta aquí resultaría suficiente.

A partir de este momento, pueden utilizarse las utilidades `ldapadd` para añadir entradas, `ldapmodify` y `ldapmodrdn` para modificar entradas o nombres relativos de entrada<sup>3</sup>, `ldapdelete` para eliminar entradas, `ldappasswd` para modificar la contraseña de una entrada y la ya mencionada `ldapsearch` para buscar entradas, desde cualquiera de los clientes. Se recomienda visitar las páginas de manual correspondientes para más información.

---

<sup>3</sup>El nombre distinguido relativo de una entrada es el primer campo del nombre distinguido de dicha entrada.

Es importante recordar que las órdenes de añadir y modificar entradas esperan recibir la información en formato LDIF. Por ejemplo, para añadir una entrada de grupo denominada “alumnos” con GID 1000 deberíamos crear primeramente un archivo (`alumnos.ldif`) con el siguiente contenido (y al menos una línea en blanco al final):

```
dn: cn=alumnos,ou=Group,dc=admon,dc=com
objectclass: posixGroup
objectclass: top
cn: alumnos
userPassword: {crypt}x
gidNumber: 1000
```

Y posteriormente añadirlo al directorio mediante la orden:

```
bash# ldapadd -x -D "cn=root,dc=admon,dc=com"
-W -f alumnos.ldif
```

Evidentemente, esto resulta muy tedioso y es fácil equivocarse. Por este motivo, se recomienda la utilización de herramientas gráficas que faciliten la labor de crear, modificar y borrar entradas en el directorio. La Sección 7.2.5 amplía este aspecto.

## Configuración de OpenLDAP con Múltiples Servidores

Si las necesidades de nuestra red y/o de nuestro directorio hacen aconsejable mantener más de un servidor LDAP (por ejemplo, para poder equilibrar la carga de las consultas, y por aspectos de tolerancia a fallos) podemos configurar varios servidores LDAP que mantengan imágenes sincronizadas de la información del directorio.

Para mantener el directorio replicado en diferentes servidores, OpenLDAP propone un esquema de maestro único y múltiples esclavos. Este esquema funciona de la siguiente forma: cada vez que se produce un cambio en el directorio del servidor maestro, el servicio `slapd` escribe dicho cambio, en formato LDIF, en un fichero local de registro (es decir, log file o cuaderno de bitácora). El servidor maestro inicia otro servicio denominado `slurpd` que, cada cierto tiempo se activa, lee dichos cambios e invoca las operaciones de modificación correspondientes en todos los esclavos. En cambio, si un esclavo recibe una operación de modificación directa por parte de un cliente, ésta debe redirigirse automáticamente al servidor maestro.

Evidentemente, este esquema sólo funciona si todos los servidores (maestro y esclavos) parten de un estado del directorio común. Por ese motivo, es necesario copiar manualmente la base de datos del directorio (es decir, el contenido del directorio `/var/lib/openldap` del servidor maestro) a cada esclavo, estando los servicios `slapd` parados en todos ellos, por supuesto.

La configuración del servidor maestro y de cada esclavo sería la siguiente:

**a) Servidor maestro.** En el archivo de configuración `/etc/openldap/slapd.conf`

hay que añadir las siguientes líneas por cada servidor esclavo:

```
replica    host=esclavo.admon.com:389
           binddn="cn=Replicator,dc=admon,dc=com"
           bindmethod=simple
           credentials=CONTRASEÑA_PLANA
```

Y además hay que decirle a slapd en qué fichero de registro tiene que escribir los cambios:

```
replogfile /var/lib/openldap/master-slapd.replog
```

- b) Servidor esclavo.** Por una parte, en el servidor esclavo hay que configurar el archivo `/etc/openldap/slapd.conf` de la misma forma que en el maestro (ver Sección 7.2.2), exceptuando las líneas expuestas en el apartado anterior, que sólo corresponden al maestro.

Por otra parte, hay que incluir en dicho fichero las siguientes opciones:

```
rootdn     "cn=Replicator,dc=admon,dc=com"
updatedn    "cn=Replicator,dc=admon,dc=com"
updateref   ldap://maestro.admon.com
```

La opción `updatedn` indica la cuenta con la que el servicio `slurpd` del servidor maestro va a realizar las modificaciones en la réplica del esclavo. Como puede comprobarse, hemos establecido que esta cuenta sea también el “rootdn” del servidor esclavo. Esa es la forma más sencilla de asegurar que este usuario tendrá permisos para modificar el directorio en este servidor (si no fuera así, deberíamos asegurarnos que esta cuenta tuviera concedido permiso de escritura en el directorio del servidor esclavo, en directiva “access” correspondiente). Por su parte, `updateref` indica al servidor esclavo que cualquier petición directa de modificación que venga de un cliente debe ser redireccionada al servidor maestro del directorio.

#### 7.2.4. Implementación de un Dominio Linux con OpenLDAP

Una vez configurado el servidor de LDAP para almacenar la información del directorio, y los clientes de LDAP para poder preguntar por ella, el siguiente y último paso para organizar un dominio Linux con LDAP es conseguir que el directorio sea utilizado por todos los ordenadores (servidores y clientes) como fuente de información administrativa, añadida a los propios ficheros de configuración locales presentes en cada ordenador.

En principio, la información administrativa que tiene sentido centralizar en un dominio Linux se reduce prácticamente a cuentas de usuario (incluyendo contraseñas) y cuentas de grupo<sup>4</sup>. En conjunto, la información almacenada en ambos tipos de cuentas permite autenticar a un usuario cuando éste desea iniciar una sesión interactiva

---

<sup>4</sup>Las cuentas de ordenadores del directorio, es decir, la centralización del fichero `/etc/hosts` del servidor LDAP, pueden obviarse si ya disponemos de resolución de nombres basada en DNS.

en un sistema Linux y, en el caso de que la autenticación sea positiva, crear el contexto de trabajo inicial (es decir, el proceso *shell* inicial) para ese usuario. Manteniendo ambos tipos de cuentas en el directorio permitiría una gestión completamente centralizada de los usuarios del dominio.

Internamente, este proceso de autenticación y creación del contexto inicial que Linux lleva a cabo cuando un usuario desea iniciar una sesión interactiva utiliza dos bibliotecas distintas:

- a) **PAM** (Pluggable Authentication Module) es una biblioteca de autenticación genérica que cualquier aplicación puede utilizar para validar usuarios, utilizando por debajo múltiples esquemas de autenticación alternativos (ficheros locales, Kerberos, LDAP, etc.). Esta biblioteca es utilizada por el proceso de “login” para averiguar si las credenciales tecleadas por el usuario (nombre y contraseña) son correctas.
- b) **NSS** (Name Service Switch) presenta una interfaz genérica para averiguar los parámetros de una cuenta (como su UID, GID, *shell* inicial, directorio de conexión, etc.), y es utilizada por el proceso de “login” para crear el proceso de atención inicial del usuario.

La ventaja fundamental de ambas bibliotecas consiste en que pueden reconfigurarse dinámicamente mediante ficheros, sin necesidad de recompilar las aplicaciones que las utilizan. Por tanto, lo único que necesitamos es reconfigurar ambas para que utilicen el servidor LDAP además de los ficheros locales (*/etc/passwd*, etc.) de cada ordenador.

En RedHat Linux, esta configuración es muy sencilla. Primero instalamos (si no lo está ya) un paquete denominado `nss_ldap`, que contiene los módulos de LDAP para PAM y NSS. A partir de ahí, ejecutamos la herramienta de configuración `authconfig`, que configura automáticamente los ficheros de PAM y NSS con los mecanismos de autenticación disponibles. En nuestro caso, basta con indicar, en dos ventanas sucesivas, que nuestro método de obtención de información y de autenticación está basado en LDAP, indicando la dirección IP del servidor y el sufijo del directorio (en formato LDAP, claro). En principio, no debemos activar la casilla de “Utilizar TLS” (que activaría conexiones seguras) ya que no hemos activado este tipo de conexiones en el servidor.

Es importante recordar que debemos realizar esta configuración en todos los clientes primero, y sólo iniciarla en el servidor cuando hayamos asegurado que todo funciona correctamente. En cualquier caso, no resulta imprescindible configurar el servidor como cliente, si siempre incluimos los usuarios y grupos nuevos tanto en el directorio como en los ficheros locales del mismo (o bien si dichos usuarios nunca van a trabajar en el servidor).

La comprobación desde cualquier cliente que el dominio funciona es muy sencilla.

Simplemente ejecutamos:

```
getent passwd  
getent group
```

Y comprobamos que nos devuelve la lista completa de usuarios y grupos del servidor. En realidad, esta forma comprobaría únicamente el funcionamiento correcto de NSS sobre LDAP. Para una comprobación completa (PAM+NSS), la manera más efectiva es intentar iniciar una sesión en un cliente con un usuario que sólo esté definido en el servidor. Hay que recordar que el directorio de conexión del usuario debe existir en el ordenador cliente (localmente, o bien montado por NFS).

### 7.2.5. Herramientas Gráficas de Administración

Entre las diferentes herramientas gráficas con las que se puede manipular un directorio de tipo LDAP en Linux, hemos elegido una denominada “Directory Administrator” ([diradmin.open-it.org](http://diradmin.open-it.org)) por su sencillez y comodidad.

Una vez instalado, invocamos su ejecutable (`directory_administrator`). La primera vez que lo ejecutamos, lanza un asistente que nos pide la información sobre el servidor de LDAP, la base del directorio, una credencial para conectarnos (“dn” y contraseña), etc. Una vez terminado el asistente, debería aparecer la ventana principal, donde vemos un objeto por cada usuario y grupo dado de alta en el directorio. Una muestra de su interfaz la tenemos en la Figura 7.2.

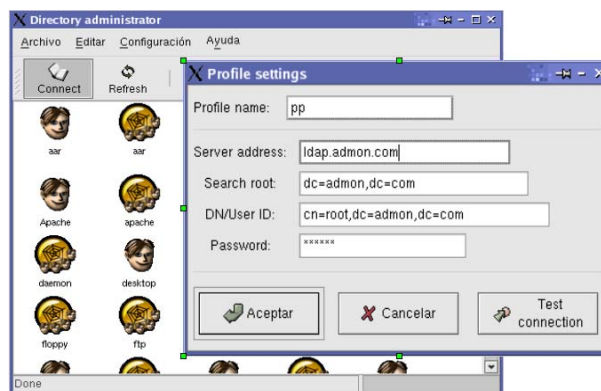


Figura 7.2: Vista de la herramienta Directory Administrator.

Antes de seguir, tenemos que ir a las opciones de configuración para activar la codificación de contraseñas “md5” y para desactivar el uso del atributo “authPassword” para almacenar la contraseña.

A partir de ese momento, ya estamos en condiciones de añadir, modificar y borrar usuarios y grupos del directorio, mediante una simple interacción con la herramienta. Si esta herramienta se convierte en la herramienta fundamental de gestión de usuarios

y grupos en el dominio Linux, necesitamos tener en mente un par de precauciones: en primer lugar, si deseamos mantener la filosofía de grupos privados, debemos crear el grupo privado de un usuario antes que el propio usuario, ya que en este último paso sólo podemos seleccionar su grupo primario. Y en segundo lugar, tendremos que gestionar manualmente los directorios de conexión de los usuarios que creemos.

### 7.3. Network File System (NFS)

Tal como se explica en el Capítulo 6 un sistema Linux puede trabajar únicamente con una jerarquía de directorios, de tal forma que si se desea acceder a distintos sistemas de archivos (particiones de discos, cd-roms, disquetes, etc.), todos ellos deben montarse primero en algún punto de dicha jerarquía.

Siguiendo la misma filosofía, Network File System (NFS) es un servicio de red que permite a un ordenador cliente montar y acceder a un sistema de archivos (en concreto, un directorio) remoto, exportado por otro ordenador servidor. Por ejemplo, supóngase que una máquina denominada *faemino* desea acceder al directorio */home/ftp/pub* de la máquina *cansado*. Para ello, debería invocarse el siguiente mandato (en *faemino*):

```
mount -t nfs cansado:/home/ftp/pub /mnt
```

Este mandato indica que el directorio */home/ftp/pub* que es exportado por el ordenador *cansado* debe ser montado en el directorio local */mnt* de *faemino*. Este directorio local debe existir previamente. La opción *-t nfs* indica a *mount* el tipo de sistema de archivos que va a montar, aunque en este caso podría omitirse, ya que *mount* detecta por el carácter *:* que se trata de un montaje remoto.

Una vez el montaje se ha realizado, cualquier acceso a archivos o directorios dentro de */mnt* (lectura, escritura, cambio de directorio, etc.) se traduce de forma transparente a peticiones al ordenador servidor (*cansado*), que las resolverá y devolverá su respuesta al cliente, todo a través de la red. Es decir, el montaje de directorios mediante NFS permite trabajar con archivos remotos exactamente igual que si fueran locales, aunque lógicamente con una menor velocidad de respuesta.

En general, NFS es muy flexible, y admite diversos escenarios:

- Un servidor NFS puede exportar más de un directorio y atender simultáneamente a varios clientes.
- Un cliente NFS puede montar directorios remotos exportados por diferentes servidores.
- Cualquier sistema UNIX puede ser a la vez cliente y servidor NFS.

Teniendo esto en cuenta, existen varios usos típicos de NFS donde este servicio muestra su utilidad:



- a) **Directorios de conexión (home) centralizados.** Cuando en una red local de máquinas Linux se desea que los usuarios puedan trabajar indistintamente en cualquiera de ellas, es apropiado ubicar los directorios de conexión de todos ellos en una misma máquina y hacer que las demás monten esos directorios mediante NFS.
- b) **Compartición de directorios de uso común.** Si varios usuarios (desde distintas máquinas) trabajan con los mismos archivos (de un proyecto común, por ejemplo) también resulta útil compartir (exportar+montar) los directorios donde se ubican dichos archivos.
- c) **Ubicar software en un solo ordenador de la red.** Es posible instalar software en un directorio del servidor NFS y compartir dicho directorio vía NFS. Configurando los clientes NFS para que monten dicho directorio remoto en un directorio local, este software estará disponible para todos los ordenadores de la red.

### 7.3.1. Cómo Funciona NFS

El funcionamiento de NFS está basado, por una parte, en dos servicios de red presentes en el ordenador servidor (`mountd` y `nfsd`) y por otra en la capacidad del cliente de traducir los accesos de las aplicaciones a un sistema de archivos en peticiones al servidor correspondiente a través de la red. Esta funcionalidad del cliente se encuentra normalmente programada en el núcleo de Linux, por lo que no necesita ningún tipo de configuración.

Respecto al servidor, el servicio `mountd` se encarga de atender las peticiones remotas de montaje, realizadas por la orden `mount` del cliente (como la del ejemplo anterior). Entre otras cosas, este servicio se encarga de comprobar si la petición de montaje es válida y de controlar bajo qué condiciones se va a acceder al directorio exportado (sólo lectura, lectura/escritura, etc.). Una petición se considera válida cuando el directorio solicitado ha sido explícitamente exportado y el cliente tiene permisos suficientes para montar dicho directorio. Esto se detalla más adelante en el documento. Una vez un directorio remoto ha sido montado con éxito, el servicio `nfsd` se dedica a atender y resolver las peticiones de acceso del cliente a archivos situados en el directorio.

### 7.3.2. Instalación y Configuración del Cliente NFS

Como se ha expresado anteriormente, el cliente NFS no requiere ni instalación ni configuración. Los directorios remotos pueden importarse utilizando el mandato `mount` y el fichero asociado `/etc/fstab`, según se muestra en el Capítulo 6.

Recuérdese que en cada invocación al mandato `mount` (y/o en cada línea del fichero `/etc/fstab`) se pueden establecer opciones de montaje. En ellas se particulariza el comportamiento que tendrá el sistema de archivos una vez se haya montado en el directorio correspondiente. En el caso de NFS, las opciones más importantes son las

que gobiernan el modo de fallo de las peticiones remotas, es decir, cómo se comporta el cliente cuando el servidor no responde:

- a) `soft`. Con esta opción, cuando una petición no tiene respuesta del servidor el cliente devuelve un código de error al proceso que realizó la petición. El problema es que muy pocas aplicaciones esperan este comportamiento y ello puede provocar situaciones en las que se pierda información. Por tanto, no es aconsejable.
- b) `hard`. Mediante esta opción, cuando el servidor no responde el proceso que realizó la petición en el cliente se queda suspendido indefinidamente. Esta es la opción que se recomienda normalmente, por ser la que esperan las aplicaciones, y por tanto más segura desde el punto de vista de los datos.

Se puede combinar con la opción `intr`, que permite matar el proceso mediante el envío de una señal (de la forma tradicional en Linux).

A continuación se muestra un ejemplo, en el que se presenta la línea del archivo `/etc/fstab` (de la máquina `faemino`) relacionada con el ejemplo de la Sección 7.3:

```
#device          mountpoint  fs-type  options  dump  fsckorder
...
cansado:/home/ftp/pub /mnt      nfs      defaults  0      0
```

### 7.3.3. Instalación y Configuración del Servidor NFS

El servidor necesita, además de los dos servicios `mountd` y `nfsd` (ambos se aúnan en el servicio común denominado `nfs`), uno más denominado `portmap` (del inglés `portmapper`), sobre el cual ambos basan su funcionamiento. Por tanto, la configuración de un servidor NFS necesita únicamente tener disponibles dichos servicios e iniciarlos en el nivel de ejecución 3 (o 5, o ambos) de la máquina (ver Capítulo 3).

Una vez activos los servicios NFS, el servidor tiene que indicar explícitamente qué directorios desea que se exporten, a qué máquinas y con qué opciones. Para ello existe un fichero de configuración denominado `/etc/exports`. A continuación se muestra uno de ejemplo, sobre el que se explican las características más relevantes:

```
# Directory          Clients and (options)
/tmp                 pc02???.dsic.upv.es(rw) *.disca.upv.es()
/home/ftp/pub       158.42.54.1(rw,root_squash)
/                   mio.dsic.upv.es(rw,no_root_squash)
/pub                (rw,all_squash,anonuid=501,anongid=601)
/pub/nopublic       (noaccess)
```

Cada línea especifica un directorio que se va a exportar, junto con una lista de autorizaciones, es decir, qué ordenadores podrán montar dicho directorio y con qué opciones (desde el punto de vista del servidor). Cada elemento de la lista de ordenadores

puede especificar un solo ordenador (mediante nombre simbólico o dirección IP) o un grupo de ordenadores (mediante el uso de caracteres comodín como '\*' ó '?'). Cuando el ordenador/rango no se especifica (por ejemplo, en las últimas dos líneas), esto significa que el directorio correspondiente se exporta a todos los ordenadores del mundo (conectados a Internet). Por su parte, de entre las posibles opciones de montaje que, entre paréntesis, pueden especificarse para cada ordenador/grupo, las más importantes se resumen en la Tabla 7.1.

Opción	Significado
( )	Esta opción establece las opciones que NFS asume por defecto.
ro	Sólo lectura.
rw	Lectura/escritura.
root_squash	Los accesos desde el cliente con UID=0 (root) se convierten en el servidor en accesos con UID de un usuario anónimo (opción por defecto).
no_root_squash	Se permite el acceso desde un UID = 0 sin conversión. Es decir, los accesos de root en el cliente se convierten en accesos de root en el servidor.
all_squash	Todos los accesos se transforman en accesos de usuario anónimo.
anonuid, anongid	Cuando se activa la opción <code>root_squash</code> ó <code>all_squash</code> , los accesos anónimos utilizan normalmente el usuario <code>nobody</code> , si existe en el servidor (defecto). Estos dos parámetros establecen qué <code>uid</code> y <code>gid</code> tendrá la cuenta anónima que el servidor utilizará para acceder contenido del directorio.
noaccess	Impide el acceso al directorio especificado. Esta opción es útil para impedir que se acceda a un subdirectorio de un directorio exportado.

**Cuadro 7.1:** Opciones más usuales en la exportación de directorios mediante NFS.

Es importante destacar que cada vez que se modifica este fichero, para que se activen los cambios, el servidor NFS debe ser actualizado ejecutando el mandato `exportfs -ra`.

Una lista completa de las opciones de montaje y su significado pueden encontrarse en la página de manual `exports (5)` de Linux. La mayoría de estas opciones establecen como quién se comporta el proceso cliente cuando su petición de acceso llega al servidor. En principio, cada petición lleva asociada el UID y GID del proceso cliente, es decir, se comporta como sí mismo. No obstante, si está activa la opción `all_squash` (o bien el UID es cero y `root_squash` está activado), entonces el UID/GID se convierten en los del usuario anónimo. De todas formas, hay que tener en cuenta que los permisos sobre cada acceso del cliente se evalúan mediante los UID/GID que finalmente son válidos en el servidor (es decir, los originales o los anónimos, según el caso).

Es muy importante resaltar que no existe ningún proceso de acreditación de usuarios en NFS, por lo que el administrador debe decidir con cautela a qué ordenadores

exporta un determinado directorio. Un directorio sin restricciones es exportado, en principio, a cualquier otro ordenador conectado con el servidor a través de la red (incluida Internet). Si en un ordenador cliente NFS existe un usuario con un UID igual a *X*, este usuario accederá al servidor NFS con los permisos del usuario con el UID igual a *X del servidor*, aunque se trate de usuarios distintos.

# Capítulo 8

## Dominios en Windows 2000

### Índice General

---

<b>8.1. Concepto de Dominio en Windows 2000 . . . . .</b>	<b>83</b>
<b>8.2. Implementación de Dominios: el Directorio Activo . . . . .</b>	<b>85</b>
8.2.1. Concepto de Directorio . . . . .	85
8.2.2. Concepto de Dominio Según el Directorio Activo . . . . .	86
8.2.3. Múltiples Dominios en la Misma Organización . . . . .	86
<b>8.3. Principales Objetos que Administra un Dominio . . . . .</b>	<b>87</b>
8.3.1. Usuarios Globales . . . . .	87
8.3.2. Grupos Globales . . . . .	88
8.3.3. Equipos . . . . .	89
8.3.4. Unidades Organizativas . . . . .	90
<b>8.4. Compartición de Recursos entre Sistemas 2000 . . . . .</b>	<b>91</b>
<b>8.5. Confianzas entre Dominios Windows 2000 . . . . .</b>	<b>92</b>
<b>8.6. Mandatos Windows 2000 para Compartir Recursos . . . . .</b>	<b>93</b>
<b>8.7. Delegación de la Administración a Unidades Organizativas . . . . .</b>	<b>94</b>

---



## 8.1. Concepto de Dominio en Windows 2000

Hoy en día, los ordenadores existentes en cualquier organización se encuentran muy frecuentemente formando parte de redes de ordenadores. En una red, los ordenadores se encuentran interconectados, de forma que pueden intercambiar información. No es necesario que un ordenador con Windows 2000 participe de una red. Sin embargo, si desea hacerlo, tiene que ser de alguna de las dos formas siguientes:

- a) **En un grupo de trabajo** (workgroup). Un grupo de trabajo es una agrupación lógica de máquinas, sin ningún otro fin. Los ordenadores que forman parte del mismo grupo aparecen juntos cuando se explora el “Entorno de Red” (o red de ordenadores NetBIOS). En este caso, la administración de cada ordenador es local e independiente del resto. Para poder acceder a los recursos que exporta un ordenador concreto, el usuario remoto tiene que disponer necesariamente de una cuenta en dicho ordenador, y permisos suficientes sobre el recurso que se comparte.
- b) **Formando parte de un dominio** (domain). Es la forma en que se recomienda que se defina una red de máquinas W2000. En un dominio, la información administrativa se encuentra centralizada, y por ello resulta más fácil y segura de gestionar. Todo este tema se centrará en definir y exponer los conceptos relacionados con los dominios en Windows 2000.

Intuitivamente, se puede definir *dominio* como una agrupación lógica de servidores de red y otros ordenadores, que comparten información común sobre cuentas (de usuarios, grupos, equipos, etc.) y seguridad. En el apartado siguiente veremos una definición más formal y completa de dominio.

Es importante matizar que el término dominio no incluye ninguna información acerca de la ubicación de los ordenadores. En realidad, no es necesario que los ordenadores que forman un dominio se encuentren físicamente cercanos, ni que estén interconectados mediante una red de algún tipo específico. De hecho, las máquinas que forman un dominio pueden estar conectadas a través de una red de área amplia o WAN (Wide Area Network), aunque lo más normal es que formen una red de área local o LAN (Local Area Network). En general, Windows 2000 separa la agrupación lógica de ordenadores, definida mediante el concepto de *dominio*, de su agrupación física o topológica, definida mediante los conceptos de *subred* y *sitio*. Una subred es un conjunto de ordenadores físicamente conectados a una red de área local. Un sitio está formado por una o varias subredes que se encuentran “bien conectadas”. Este término hace referencia a que la velocidad de interconexión entre dichas subredes es “suficiente” (a criterio del administrador) para dar soporte al tráfico de la información del dominio concreto ubicado en dichas subredes. Un dominio W2000 puede abarcar uno o varios de estos *sitios*.

Por otra parte, es en el ámbito de las redes de sistemas W2000 donde cobra sentido el hecho de que existan dos versiones de este sistema operativo: Windows 2000 Server y Windows 2000 Professional. Habitualmente, un Windows 2000 Server (o servidor

W2000) tiene un papel especial en una red W2000, y se utiliza para proporcionar servicios centralizados de red en el dominio. Por su parte, un Windows 2000 Professional (o estación W2000) es una máquina que, aunque puede formar parte de un dominio, no proporciona ningún servicio al resto de máquinas, siendo una estación de trabajo de propósito general.

Concretamente, en un dominio de Windows 2000:

- Existe *necesariamente* un servidor W2000 con funciones de Controlador de Dominio (Domain Controller, o DC). Entre otros servicios, este servidor posee información centralizada sobre los recursos que posee el dominio y puede administrar los ordenadores que pertenecen al mismo.
- Pueden existir otros servidores W2000 con funciones de Controladores de Dominio, de forma que todos ellos replican la información de los recursos del dominio y se reparten la carga de proporcionar información y los servicios de administración. En contraposición a lo que ocurría en Windows NT 4.0, en Windows 2000 no existen controladores principales ni secundarios de dominio, sino que todos se encuentran en el mismo nivel de jerarquía. Sin embargo, sí pueden distribuirse entre ellos los distintos servicios que puede proporcionar un DC.
- Pueden existir uno o varios servidores W2000 sin funciones especiales de administración dentro del dominio. A estos ordenadores se les llama servidores miembro (member servers). Habitualmente estas máquinas proporcionan algún servicio específico dentro del dominio (servicios de impresión, servicios de acceso a datos, servidores web, etc.), pero no guardan ninguna información administrativa.
- Pueden existir una o varias estaciones W2000, que se utilizarán para trabajo habitual.

Tanto los servidores miembro como las estaciones W2000 son *clientes* de los DCs respecto a la información relativa a usuarios, grupos, contraseñas y autenticación, etc. A diferencia de Windows NT 4.0, en Windows 2000 los DCs se instalan siempre como servidores miembro inicialmente y luego, en su caso, se *promocionan* a controlador de dominio (de un dominio existente o de un nuevo dominio). Esta promoción se lleva a cabo mediante la ejecución de la utilidad DCPROMO.

Finalmente, es importante saber que un dominio Windows 2000 puede encontrarse configurado en uno de dos *modos* alternativos: *modo mixto* y *modo nativo*. El modo mixto es el modo en el que los DCs se promocionan por defecto. Este modo ofrece compatibilidad (a nivel de controlador) a sistemas NT anteriores, y resulta por tanto necesario si alguno de los controladores del dominio es un sistema NT 4.0. Por contra, si todos los DCs del dominio son sistemas Windows 2000, podemos configurar el dominio en modo nativo. Como veremos, existen ciertas funcionalidades en un dominio Windows 2000 que sólo están disponibles en modo nativo. Un dominio en modo mixto puede pasarse a modo nativo mediante una acción del administrador. Sin embargo, la acción contraria no es posible (es decir, el paso de modo mixto a nativo es irreversible).



## 8.2. Implementación de Dominios: el Directorio Activo

### 8.2.1. Concepto de Directorio

Tal como hemos visto, el concepto intuitivo de dominio es el de la centralización de la información (y las labores) de administración de una red de ordenadores, de forma que la gestión de dichos ordenadores resulte más cómoda y eficiente. Windows 2000 implementa el concepto de dominio mediante un concepto más básico, el de *directorio*.

En el ámbito de las redes de ordenadores, un directorio (o almacén de datos) es una estructura jerárquica que almacena información sobre recursos (o de forma más general, *objetos*) en la red. El directorio se implementa normalmente como una base de datos optimizada para operaciones de lectura (soporta búsquedas de grandes cantidades de información) y con capacidades de exploración.

En concreto, el servicio de directorio que incorpora Windows 2000 se denomina *Directorio Activo* (Active Directory). El Directorio Activo es un servicio de red que almacena información acerca de los recursos existentes en la red y controla el acceso de los usuarios y las aplicaciones a dichos recursos. De esta forma, se convierte en un medio de organizar, administrar y controlar *centralizadamente* el acceso a los recursos de la red.

El Directorio Activo se ha implementado siguiendo una serie de estándares y protocolos existentes, ofreciendo interfaces de programación de aplicaciones que facilitan la comunicación con otros servicios de directorio. Entre ellos, se pueden encontrar los siguientes:

- DHCP (Dynamic Host Configuration Protocol): Protocolo de configuración dinámica de ordenadores, que permite la administración desatendida de direcciones de red.
- DNS (Domain Name System): Servicio de nombres de dominio que permite la administración de los nombres de ordenadores. Este servicio constituye el mecanismo de asignación y resolución de nombres (traducción de nombres simbólicos a direcciones IP) en Internet.
- SNTP (Simple Network Time Protocol): Protocolo simple de tiempo de red, que permite disponer de un servicio de tiempo distribuido.
- LDAP (Lightweight Directory Access Protocol): Protocolo ligero (o compacto) de acceso a directorio. Este es el protocolo mediante el cual las aplicaciones acceden y modifican la información existente en el directorio.
- Kerberos V5: Protocolo utilizado para la autenticación de usuarios y máquinas.
- Certificados X.509: estándar que permite distribuir información a través de la red de una forma segura.

### 8.2.2. Concepto de Dominio Según el Directorio Activo

Desde el punto de vista del Directorio Activo, podemos ahora definir con más propiedad el concepto de *dominio* en Windows 2000: un dominio es un conjunto de equipos que comparten una base de datos de directorio común y que se identifica mediante un nombre de dominio DNS.

En una red Windows 2000, un dominio define:

- a) **Límite de seguridad.** El administrador de un dominio posee los permisos y derechos necesarios para administrar los recursos de ese dominio únicamente (a menos que se le hayan concedido de forma explícita en otros dominios). Es decir, el dominio marca los límites de la administración (y de la seguridad).
- b) **Unidad de replicación.** Todos los controladores de dominio (DCs) de un dominio poseen una copia completa de la información de directorio de dicho dominio. Para ello, las actualizaciones de dicha información en cualquier controlador se replican de forma automática al resto.

Como se ha comentado arriba, Windows 2000 ha incorporado el esquema de nombrado DNS para nombrar a los dominios y para publicar los servicios que cada ordenador ofrece al resto dentro del dominio. Es más, existe una relación biunívoca entre un dominio Windows 2000 y un dominio DNS, independientemente de que el dominio W2000 forme parte o no de Internet. Precisamente es mediante este esquema de nombrado DNS como mejor se entiende la *jerarquía* en la que el Directorio Activo permite organizar los dominios de una organización. Esto se explica a continuación.

### 8.2.3. Múltiples Dominios en la Misma Organización

Existen muchos casos en los que es interesante disponer de varios dominios de ordenadores Windows 2000 en la misma organización (distribución geográfica o departamental, distintas empresas, etc.). El Directorio Activo permite almacenar y organizar la información de directorio de varios dominios de forma que, aunque la administración de cada uno sea independiente, dicha información esté disponible para todos los dominios.

En concreto, los dominios de Windows 2000 se pueden organizar en dos unidades jerárquicas:

- a) **Arboles.** Un árbol es una jerarquía de dominios que comparten un sufijo DNS. El dominio situado en la *raíz del árbol* se denomina principal y los posibles subdominios que se creen por debajo se denominan secundarios. Normalmente, al menos un controlador de dominio de un dominio principal es un servidor DNS. Por ejemplo, si en la UPV se quisiera tener un dominio Windows 2000 por departamento, debería existir un dominio principal denominado `upv.es` (en este caso, la raíz del árbol) y tantos dominios secundarios como departamentos. En este caso, el dominio del departamento DSIC debería denominarse `dsic.upv.es`,

ya que ese es el dominio DNS correspondiente a las máquinas del departamento. Si dentro del dominio del DSIC se quisieran crear subdominios W2000, tendrían que crearse necesariamente los subdominios DNS correspondientes en el servidor DNS de la universidad.

- b) **Bosques.** Pongámonos ahora en el caso de que no todos los dominios de una organización compartan el mismo sufijo DNS. En este caso, cada agrupación de dominios con el mismo sufijo DNS formarían un árbol. Según la organización del Directorio Activo, el conjunto de dichos árboles puede constituir una unidad jerárquica superior que se denomina (lógicamente) bosque.

La información del directorio es accesible para todo el bosque de dominios. De hecho, la parte fundamental del directorio (denominada *esquema*) que define los tipos de objetos y atributos que se pueden crear en el directorio es única para todo el bosque. Ello asegura que la información que se almacena en la parte del directorio de cada dominio del bosque es homogénea.

En resumen, cuando promocionamos un servidor W2000 a controlador de dominio, tenemos que decidir una de las siguientes opciones de instalación:

- a) DC adicional de un dominio existente o DC de un dominio nuevo.
- b) En el segundo caso, el dominio (nuevo) puede ser un dominio secundario de otro dominio existente (es decir, un subdominio de un árbol de dominios ya creado), o bien el dominio principal (raíz) de un nuevo árbol de dominios.
- c) En este segundo caso, el dominio raíz puede ser de un bosque existente o de un nuevo bosque.

## 8.3. Principales Objetos que Administra un Dominio

### 8.3.1. Usuarios Globales

En el Capítulo 5 se ha visto cómo en Windows 2000 pueden crearse cuentas de usuarios y grupos, y cómo se utilizan ambas para, por una parte, identificar y autenticar a las personas que pueden acceder al sistema, y por otra, administrar los permisos y derechos que controlarán el acceso de dichas personas en dicho sistema. Por lo tanto, si una persona tiene que trabajar en varios ordenadores, debe en principio poseer una cuenta de usuario en cada uno de ellos.

En un dominio, cualquier servidor W2000 que actúa como DC puede crear cuentas de *usuario global*. En este caso, el término *global* se interpreta como global al dominio. Los datos de una cuenta de usuario global se almacenan en el Directorio Activo y son accesibles por todos los ordenadores del dominio. Es decir, no es que se cree una cuenta en cada ordenador miembro del dominio, sino que existe una única cuenta (con un único SID) que es visible en todos los ordenadores del dominio. En este caso, cuando

una persona se conecta a cualquier ordenador miembro de un dominio utilizando para ello una cuenta de usuario global, dicho ordenador realiza una consulta al Directorio Activo (i.e., a alguno de los DCs) para que se validen las credenciales del usuario. El resultado de esta validación es entonces enviado al ordenador miembro, y de éste al usuario, concediendo o rechazando la conexión.

Un ordenador miembro de un dominio (una estación W2000, por ejemplo), además de poder ver a los usuarios globales del dominio, puede crear también sus propios usuarios locales, que son únicamente visibles en dicho ordenador. Cuando una persona desea entrar en el sistema utilizando una cuenta local, dicha cuenta se valida contra la base de datos local de ese ordenador. Además, es importante notar que a dicho usuario local no se le pueden asignar permisos sobre recursos que residan en otro sistema W2000 (puesto que allí no existe). Por el contrario, a un usuario global se le pueden conceder permisos sobre cualquier recurso (archivo, directorio, impresora, etc.) de cualquier ordenador miembro del dominio, puesto que es visible (y posee el mismo SID) en todos ellos.

### 8.3.2. Grupos Globales

De forma análoga a los usuarios globales, existen *grupos* (de seguridad) que son almacenados en el Directorio Activo y que por tanto son visibles desde todos los ordenadores del dominio (y de otros dominios). En concreto, en dominios Windows 2000 se definen tres tipos de grupos globales (o, de forma más precisa, se pueden definir grupos de tres *ámbitos* distintos):

- a) **Grupos locales del dominio.** En un dominio en modo mixto, pueden contener cuentas de usuario y grupo globales de cualquier dominio del bosque. En un dominio en modo nativo, pueden contener además grupos universales y otros grupos locales del dominio. Sólo son visibles en el dominio en que se crean, y suelen utilizarse para conceder permisos y derechos en cualquiera de los ordenadores del dominio<sup>1</sup>.
- b) **Grupos globales.** En un dominio en modo mixto, pueden contener cuentas de usuario globales del mismo dominio. En un dominio en modo nativo, pueden contener además otros grupos globales del mismo dominio. Son visibles en todos los dominios del bosque, y suelen utilizarse para clasificar a los usuarios en función de las labores que realizan.
- c) **Grupos universales.** Sólo están disponibles en dominios en modo nativo. Pueden contener cuentas de usuario y grupos globales, así como otros grupos universales, de cualquier dominio del bosque. Son visibles en todo el bosque.

En un ordenador miembro de un dominio también se pueden definir grupos locales. Los grupos locales pueden estar formados por cuentas de usuario locales y usuarios y grupos globales de cualquier dominio del bosque (en modo mixto) y además por

---

<sup>1</sup>En modo mixto, sólo son visibles por los DCs del dominio, y por tanto sólo se pueden utilizar para administrar permisos y derechos en esos ordenadores.

grupos universales (en modo nativo). Un grupo local no puede ser miembro de otro grupo local. Los grupos locales pueden utilizarse para conceder permisos y derechos sólo en el equipo en que son creados.

Por tanto, la administración de la protección en cada ordenador del dominio puede realizarse mediante grupos locales del dominio o grupos locales del equipo en que reside el recurso a administrar. Por tanto, la recomendación que se hacía en el Capítulo 5 respecto a la asignación de permisos en base a grupos locales sigue siendo válida. En el caso más general, la regla que recomienda Windows 2000 es la siguiente:

- Asignar usuarios globales a grupos globales, según las labores que desempeñen en la organización.
- Incluir (usuarios y/o) grupos globales en grupos locales (del equipo o del dominio) según el nivel de acceso que vayan a tener.
- Asignar permisos y derechos exclusivamente a estos grupos locales (del equipo o del dominio).

En relación con esto, es importante saber que cuando un ordenador pasa a ser miembro de un dominio, el grupo global *Administradores del dominio* se incluye automáticamente en el grupo local *Administradores* de dicho ordenador. De igual forma, el grupo global *Usuarios* se incluye dentro del grupo local *Usuarios*. De esta forma, los administradores y usuarios normales del dominio tienen en cada miembro los mismos derechos y permisos que los que tengan ya definidos los administradores y usuarios locales, respectivamente. El administrador local puede, si lo desea, invalidar esta acción automática, extrayendo posteriormente los grupos globales de los locales.

### 8.3.3. Equipos

Como hemos visto, en el Directorio Activo de un dominio se conserva toda la información relativa a cuentas de usuarios y grupos globales. Esta misma base de datos de directorio recoge también una *cuenta de equipo* por cada uno de los ordenadores miembro de un dominio.

Entre otras informaciones, en cada una de estas cuentas se almacena el nombre del ordenador, así como un identificador único y privado que lo identifica unívocamente. Este identificador es análogo al SID de cada cuenta de usuario, y sólo lo conocen los DCs y el propio ordenador miembro. Es por tanto, un dato interno del sistema operativo, y ni siquiera el administrador puede cambiarlo.

Windows 2000 puede utilizar distintos protocolos de comunicaciones seguros entre los ordenadores miembro de un dominio y los DCs. Entre ellos los más importantes son NTLM (el protocolo utilizado por NT 4.0, que se mantiene por compatibilidad hacia atrás) y Kerberos V5. Kerberos presenta numerosas ventajas respecto a NTLM, pero sólo es viable en la práctica si todas las máquinas del dominio son Windows 2000. Estos protocolos se utilizan siempre que información relativa a aspectos de seguridad

se intercambia entre máquinas W2000 pertenecientes a algún dominio y, en concreto, para autenticar usuarios (como se ha explicado arriba).

#### 8.3.4. Unidades Organizativas

Una *unidad organizativa* es en realidad un *contenedor* de objetos del Directorio Activo, es decir, un lugar dentro del cual se pueden crear otros objetos del directorio. En concreto, dentro de una unidad de este tipo pueden crearse cuentas de usuario, de grupo, de equipo, de recurso compartido, de impresora compartida, etc., además de *otras* unidades organizativas. Los objetos creados dentro de una unidad organizativa pueden moverse más tarde a otra, siguiendo las necesidades de organización y/o administración de la empresa, tantas veces como sea necesario.

En definitiva, lo que se pretende es poder organizar una *jerarquía* de entidades en el directorio de un cierto dominio, agrupándolas de forma coherente. Esta organización permite:

- Por una parte, tener una *estructuración lógica* de los objetos del directorio, de acuerdo con la organización de la empresa (por departamentos, ubicaciones, delegaciones geográficas, etc.). Entre otras ventajas, esta organización permite localizar los objetos de forma más fácil (por ejemplo, localizar las impresoras compartidas del edificio central de la delegación de Alacant).
- Por otra parte, cada unidad organizativa puede *administrarse de forma independiente*. En concreto, se puede otorgar la administración total o parcial de una unidad organizativa a un usuario o grupo de usuarios, permitiendo así la delegación de la administración a usuarios con el nivel de responsabilidad adecuada (esto se verá en el Apartado 8.7 al final del capítulo). Además, es posible establecer *políticas de grupo* independientes a los objetos de cada unidad organizativa, lo cual permite establecer (de forma centralizada) una serie de restricciones distintas a usuarios y/o equipos de cada unidad.

En muchos sentidos, el concepto de unidad organizativa se puede utilizar en Windows 2000 de la misma forma que se entendía el concepto de dominio en NT 4.0, es decir, conjunto de usuarios, equipos y recursos administrados independientemente. Como se ha visto anteriormente, en Windows 2000 el concepto de dominio viene más bien asociado a la distribución de los sitios y a la implementación de DNS que exista (o quiera crearse) en la empresa.

De este modo, existen muchos casos en los que, en vez de crear múltiples dominios (por motivos de administración), resulta más conveniente implementar un modelo de dominio único y crear múltiples unidades organizativas, estableciendo la administración de forma adecuada en cada una de ellas.

## 8.4. Compartición de Recursos entre Sistemas 2000

Cuando un sistema W2000 participa en una red (grupo de trabajo o dominio), puede compartir sus recursos con el resto de ordenadores. En este contexto, sólo vamos a considerar como recursos a compartir los directorios o carpetas que existen en un sistema W2000<sup>2</sup>.

Cualquier sistema W2000 puede compartir carpetas, tanto si es un servidor como si es una estación de trabajo. Para poder compartir una carpeta, basta con desplegar su menú contextual desde una ventana o desde el explorador de archivos, y seleccionar *Compartir...* En la ventana asociada a esta opción se determina el nombre que tendrá el recurso (que no tiene por qué coincidir con el nombre de la carpeta), así como qué usuarios van a poder acceder al mismo y con qué permisos. En relación con esto, existe una gran diferencia entre que la carpeta resida en una partición FAT y que resida en una NTFS, como se explica a continuación.

Si la carpeta reside en una partición FAT, este filtro de acceso será el único que determine los usuarios que van a poder acceder al contenido del directorio, puesto que no es posible determinar permisos sobre la propia carpeta o sus archivos. Es decir, el filtro sólo se establece para poder acceder al recurso. Si un usuario tiene permisos suficientes para conectarse a un recurso, tendrá acceso sobre todos los archivos y subcarpetas del recurso. El tipo de acceso sobre todos ellos será el que le permita el permiso sobre el recurso (Lectura, Escritura, Control Total, etc.).

Por el contrario, si la carpeta se encuentra en una partición NTFS, ésta tendrá unos permisos establecidos (así como sus subcarpetas y archivos), al margen de ser una carpeta compartida. En este último caso también es posible establecer un filtro desde la ventana de *Compartir...*, pero entonces sólo los usuarios que puedan pasar ambos filtros podrán acceder a la carpeta compartida. En este caso se recomienda dejar Control Total sobre Todos en los permisos asociados al recurso (opción por defecto), y controlar quién (y cómo) puede acceder al recurso y a su contenido mediante los permisos asociados a la propia carpeta y a sus archivos y subcarpetas.

Esta recomendación es muy útil, si tenemos en cuenta que de esta forma para cada carpeta (y archivo) del sistema no utilizamos dos grupos de permisos sino uno solo, independientemente de que la carpeta esté compartida o no. Este forma de trabajar obliga al administrador a asociar los permisos correctos a cada objeto del sistema (aunque no esté compartido), pero por otra parte unifica la visión de la seguridad de los archivos, con lo que a la larga resulta más segura y más sencilla.

En cualquiera de los dos casos anteriores (partición FAT o NTFS), hay que tener en cuenta además que si un usuario ha iniciado una sesión interactiva en un ordenador W2000 denominado A, y desea conectarse a un recurso de red que exporta otro W2000 denominado B, además de poseer suficientes permisos (sobre el recurso, sobre la propia carpeta y sobre su contenido), tiene que tener concedido en B el derecho *Acceder a este equipo desde la red*.

---

<sup>2</sup>La compartición de otros recursos (tales como impresoras, por ejemplo) queda fuera del ámbito de este texto.

Respecto al Directorio Activo, una vez hemos compartido físicamente un directorio en la red (según el procedimiento descrito arriba), podemos además *publicar* este recurso en el directorio. Para ello debemos crear un nuevo objeto, en la unidad organizativa adecuada, de tipo *recurso compartido*. A este objeto le debemos otorgar un nombre simbólico y su nombre de recurso en la red (\\equipo\recurso). Es importante tener en cuenta que cuando se publica el recurso, no se comprueba si realmente existe o no, por lo que es responsabilidad del administrador el haberlo compartido y que su nombre coincida con el de la publicación. Una vez publicado, el recurso puede localizarse mediante búsquedas en el Directorio Activo, como el resto de objetos del mismo.

Por último, cuando un sistema W2000 forma parte de un dominio, al margen de los recursos que el administrador desee compartir, se comparten por defecto otros recursos para usos del propio sistema y administrativos. Estos recursos no deben modificarse ni prohibirse:

- a) (letra\_de\_unidad)\$ . Por cada partición existente en un sistema W2000 (C:, D:, etc.) se crea un recurso compartido denominado C\$, D\$, etc. Los administradores del dominio, así como los operadores de copia del dominio, pueden conectarse por defecto a estas unidades.
- b) ADMIN\$. Es un recurso utilizado por el propio sistema durante la administración remota de un ordenador W2000.
- c) IPC\$. Recurso que agrupa los tubos (colas de mensajes) utilizados por los programas para comunicarse entre ellos. Se utiliza durante la administración remota de un ordenador W2000, y cuando se observa los recursos que comparte.
- d) NETLOGON. Recurso que exporta un DC para proporcionar a los ordenadores miembros del dominio el servicio de validación de cuentas globales a través de la red (Net Logon service).
- e) SYSVOL. Recurso que exporta cada DC de un dominio. Contiene información del Directorio Activo (por ejemplo, *scripts* asociados a directivas de grupo) que debe estar disponible desde todos los equipos del dominio.

En relación con los nombres de estos recursos, es interesante saber que añadir el carácter '\$' al final de cualquier nombre de recurso tiene un efecto específico: prohíbe que dicho recurso se visualice dentro de la lista de recursos que una máquina exporta al resto. Es decir, convierte un recurso en "invisible" para al resto del mundo. En este caso, un usuario remoto sólo podrá conectarse al recurso si conoce su nombre de antemano (y tiene suficientes permisos, obviamente).

## 8.5. Confianzas entre Dominios Windows 2000

Una relación de confianza (trust relationship) es una relación que une a *dos* dominios W2000, mediante la cual los usuarios (y grupos) globales de uno de ellos pueden



utilizar los recursos que exporta el segundo.

Más formalmente, se puede definir así: cuando se establece una relación de confianza entre dos dominios W2000, los usuarios y grupos globales del *dominio de confianza* (trusted domain) son visibles en los ordenadores miembro del *dominio que confía* (trusting domain), y por tanto se les puede conceder permisos sobre los recursos este último. En realidad, las relaciones de confianza entre dominios se establecen entre los DCs de ambos dominios, de forma que los DCs del dominio que confía pueden validar usuarios en los DCs del dominio de confianza.

Cuando existe una relación de confianza entre dos dominios, los usuarios del dominio de confianza aparecen en las listas de usuarios conocidos en los ordenadores del dominio que confía, y en particular cuando se trata de conceder algún derecho o algún permiso. De acuerdo con esta visión, resulta evidente que siempre existe una relación de confianza implícita entre todo miembro de un dominio W2000 y sus DCs.

Existe una diferencia fundamental entre las confianzas existentes en Windows NT 4.0 y las que implementa Windows 2000. En NT 4.0, las confianzas entre dominios debían establecerse *manualmente* entre los dominios y tenían como características fundamentales el no ser recíprocas ni transitivas. Es decir: si se deseaba que el dominio A confiara en el dominio B y que B confiara en A, era necesario establecer dos relaciones de confianza (recíprocas). Y por otra parte, si A confiaba en B, y B confiaba en C, entonces A no confiaba en C por defecto.

Por contra, en Windows 2000, las relaciones de confianzas entre dominios del mismo bosque son automáticas (se crean por defecto al añadir dominios al bosque), recíprocas y transitivas. Esta definición resulta compatible con el concepto de directorio global a todo el bosque y cumple con la visibilidad de usuarios y grupos que se ha comentado en la Sección 8.3. No obstante, en Windows 2000 también es posible establecer manualmente relaciones de confianza “al estilo NT 4.0”, para casos en los que debamos confiar en dominios NT anteriores.

## 8.6. Mandatos Windows 2000 para Compartir Recursos

La compartición de recursos en Windows 2000 puede realizarse en línea de órdenes utilizando los mandatos `net share` y `net use`. La sintaxis de ambos mandatos es la siguiente:

- Mandato `net share`

Crea, elimina o muestra recursos compartidos.

```
net share
net share recurso_compartido
net share recurso_compartido=unidad:ruta_de_acceso
    [/users:número | /unlimited] [/remark:"texto"]
net share recurso_compartido [/users:número | unlimited]
    [/remark:"texto"]
net share {recurso_compartido | unidad:ruta_de_acceso} /delete
```

#### ■ Mandato `net use`

Conecta o desconecta un equipo de un recurso compartido o muestra información acerca de las conexiones del equipo. También controla las conexiones de red persistentes.

```
net use [nombre_dispositivo]
      [\\nombre_equipo\recurso_compartido[\\volumen]]
      [contraseña | *]] [/user:[nombre_dominio\]nombre_usuario]
      [[/delete] | [/persistent:{yes | no}]]
net use nombre_dispositivo [/home[contraseña | *]] [/delete:{yes | no}]
net use [/persistent:{yes | no}]
```

## 8.7. Delegación de la Administración a Unidades Organizativas

Para delegar, total o parcialmente, la administración de una unidad organizativa existe un asistente (wizard) que aparece cuando se selecciona la acción *Delegar el control...* en el menú contextual de la unidad organizativa. Este asistente pregunta básicamente los dos aspectos propios de la delegación: *a quién* se delega y *qué* se delega. La primera pregunta se contesta o bien con un usuario o con un grupo (se recomienda un grupo). Para responder a la segunda pregunta, se puede elegir una tarea *predefinida* a delegar (de entre una lista de tareas frecuentes), o bien podemos optar por construir una tarea personalizada. En este último caso, tenemos que especificar la tarea mediante un conjunto de permisos sobre un cierto tipo de objetos del directorio. Esto se explica a continuación.

Internamente, los derechos de administración (o control) sobre un dominio o unidad organizativa funcionan de forma muy similar a los permisos sobre una carpeta o archivo: existe una DACL propia y otra heredada, que contienen como entradas aquellos usuarios/grupos que tienen concedida (o denegada) una cierta acción sobre la unidad organizativa o sobre su contenido. En este caso, las acciones son las propias de la administración de objetos en el directorio (control total, creación de objetos, modificación de objetos, consulta de objetos, etc.), donde los “objetos” son las entidades que pueden ser creados dentro de la unidad: usuarios, grupos, unidades organizativas, recursos, impresoras, etc.

En resumen, la delegación de control sobre una unidad organizativa puede hacerse de forma completa (ofreciendo el *Control Total* sobre la unidad) o de forma parcial (permitiendo la lectura, modificación y/o borrado de los objetos de la misma). Hay que tener en cuenta que en el caso de la delegación parcial, el número de posibilidades es inmenso: por una parte, se incluye la posibilidad de establecer el permiso sobre cada *atributo* de cada tipo de objeto posible; por otra parte, se puede establecer a qué unidades se va a aplicar la regla (sólo en esa unidad organizativa, en todas las que se sitúan por debajo, en parte de ellas, etc.). Por tanto, para una delegación parcial se recomienda el uso del asistente, ya que su lista de delegación de tareas más frecuentes (como por ejemplo “Crear, borrar y administrar cuentas de usuario” o “Restablecer

contraseñas en cuentas de usuario”) resulta muy útil. Sin embargo, cuando la delegación que buscamos no se encuentra en la lista, tendremos que diseñar una a medida, asignando los permisos oportunos sobre los objetos del directorio que sean necesarios.



# Capítulo 9

## Configuración de Samba

### Índice General

---

9.1. ¿Qué es samba? . . . . .	99
9.2. Configuración de Samba . . . . .	100
9.3. Niveles de Seguridad . . . . .	101
9.4. Configuración de Samba con el Nivel de Seguridad domain . . . . .	103
9.5. Tratamiento de los Accesos como Invitado . . . . .	103
9.6. El Sistema de Ficheros SMB para Linux . . . . .	104
9.7. Opciones del Servidor Samba . . . . .	105
9.8. Opciones del Recurso . . . . .	105

---



## 9.1. ¿Qué es samba?

Samba es un producto<sup>1</sup> que se ejecuta en sistemas Unix, permitiendo al sistema Unix conversar con sistemas Windows a través de la red de forma nativa. De esta forma, el sistema Unix aparece en el "Entorno de red", y clientes Windows pueden acceder a sus recursos de red e impresoras compartidas como si de otro sistema Windows se tratase. Para ello, Samba implementa los protocolos NetBIOS y SMB. NetBIOS es un protocolo de nivel de sesión que permite establecer sesiones entre dos ordenadores. SMB (Server Message Block), implementado sobre NetBIOS, es el protocolo que permite a los sistemas Windows compartir ficheros e impresoras.

Esencialmente, Samba consiste en dos programas, denominados `smbd` y `nmbd`. Ambos programas utilizan el protocolo NetBIOS para acceder a la red, con lo cual pueden conversar con ordenadores Windows. Haciendo uso de estos dos programas, Samba ofrece los siguientes servicios, todos ellos iguales a los ofrecidos por los sistemas Windows:

- Servicios de acceso remoto a ficheros e impresoras.
- Autenticación y autorización.
- Resolución de nombres.
- Anuncio de servicios.

El programa `smbd` se encarga de ofrecer los servicios de acceso remoto a ficheros e impresoras (implementando para ello el protocolo SMB), así como de autenticar y autorizar usuarios. `smbd` ofrece los dos modos de compartición de recursos existentes en Windows, basado en usuarios o basado en recursos. En el modo basado en usuarios (propio de los dominios Windows NT o 2000) la autorización de acceso al recurso se realiza en función de nombres de usuarios registrados en un dominio, mientras que en el modo basado en recursos (propio de Windows 3.11/95) a cada recurso se le asigna una contraseña, estando autorizado el acceso en función del conocimiento de dicha contraseña.

El programa `nmbd` permite que el sistema Unix participe en los mecanismos de resolución de nombres propios de Windows, lo cual incluye el anuncio en el grupo de trabajo, la gestión de la lista de ordenadores del grupo de trabajo, la contestación a peticiones de resolución de nombres y el anuncio de los recursos compartidos. De esta forma, el sistema Unix aparece en el "Entorno de Red", como cualquier otro sistema Windows, publicando la lista de recursos que ofrece al resto de la red.

Adicionalmente a los dos programas anteriores, Samba ofrece varias utilidades. Algunas de las más relevantes son las siguientes:

---

<sup>1</sup>Samba se distribuye gratuitamente para varias versiones de Unix, de acuerdo con los términos de la General Public License de GNU.

- `smbclient`. Una interfaz similar a la utilidad `ftp`, que permite a un usuario de un sistema Unix conectarse a recursos SMB y listar, transferir y enviar ficheros.
- `swat` (Samba Web Administration Tool). Esta utilidad permite configurar Samba de forma local o remota utilizando un navegador de web.
- Sistema de ficheros SMB para Linux. Linux puede montar recursos SMB en su jerarquía, al igual que sucede con directorios compartidos vía NFS.
- `winbind`. Permite integrar un servidor Samba en un dominio Windows sin necesidad de crear usuarios Unix en el servidor Samba que correspondan con los usuarios del dominio Windows, simplificando así la labor de administración.

## 9.2. Configuración de Samba

La configuración de Samba se realiza en el fichero `/etc/samba/smb.conf`. En este fichero se establecen las características del servidor Samba, así como los recursos que serán compartidos en la red. La utilización de este fichero es bastante sencilla, ya que aunque existe un gran número de opciones, muchas de ellas pueden obviarse dado que siempre existe un valor por defecto para cada opción, que suele ser apropiado. A título de ejemplo, en la Figura 9.1 se muestra un fichero de configuración simple, que exporta el directorio de conexión de cada usuario como un recurso de red distinto, y el directorio `/espacio/pub` como el recurso de red `pub`.

```
[global]
[homes]
    comment = Home Directories
[pub]
    path = /espacio/pub
```

**Figura 9.1:** Ejemplo de fichero `/etc/samba/smb.conf`

Como se ve en el ejemplo, el fichero `/etc/samba/smb.conf` se encuentra dividido en secciones, encabezados por una palabra entre corchetes. Dentro de cada sección figuran opciones de configuración, de la forma `etiqueta = valor`, que determinan las características del recurso exportado por la sección. En los apartados 9.7 y 9.8 del presente documento se citan las opciones más importantes que se pueden establecer en cada sección.

Existen tres secciones predefinidas, denominadas `global`, `homes` y `printers`, y tantas secciones adicionales como recursos extra se quieran compartir. El cometido de dichas secciones predefinidas se describe en la Tabla 9.1, a continuación.

Para cualquier otro recurso (directorio o impresora) que se quiera compartir hay que definir una sección adicional en el fichero de configuración. El encabezamiento de dicha sección (`pub` en el ejemplo anterior) corresponderá al nombre que el recurso tendrá en la red.



Sección	Cometido
[global]	Define los parámetros de Samba a nivel global del servidor, así como los parámetros que se establecerán por defecto en el resto de las secciones.
[homes]	Define automáticamente un recurso de red por cada usuario conocido por Samba. Este recurso, por defecto, está asociado al directorio de conexión de cada usuario en el ordenador en el que Samba está instalado.
[printers]	Define un recurso compartido por cada nombre de impresora conocida por Samba.

**Cuadro 9.1:** Secciones globales en el fichero `/etc/smb.conf`.

Por otra parte, Samba ofrece una interfaz de edición de este fichero basada en web denominada `swat`. Esta herramienta permite configurar Samba utilizando un navegador de red, tanto de forma local como remota. Para ello, basta con acceder a la dirección `http://nombre_de_ordenador_samba:901/` mediante cualquier navegador.

### 9.3. Niveles de Seguridad

Una de las consideraciones más importantes a la hora de configurar Samba es la selección del nivel de seguridad.

Desde la perspectiva de un cliente, Samba ofrece dos modos de seguridad, denominados `share` y `user`, al igual que sucede en los sistemas Windows:

- a) **Modo Share.** En modo `share`, cada vez que un cliente quiere utilizar un recurso ofrecido por Samba, debe suministrar una contraseña de acceso asociada a dicho recurso.
- b) **Modo User.** En modo `user`, el cliente debe establecer en primer lugar una sesión con el servidor Samba, para lo cual le suministra un nombre de usuario y una contraseña. Una vez Samba valida al usuario, el cliente obtiene permiso para acceder a los recursos ofrecidos por Samba.

En cualquiera de ambos, Samba tiene que asociar un usuario del sistema Unix en el que se ejecuta Samba con la conexión realizada por el cliente. Este usuario es el utilizado a la hora de comprobar los permisos de acceso a los ficheros y directorios que el sistema Unix/Samba comparte en la red.

La selección del nivel de seguridad se realiza con la opción `security`, la cual pertenece a la sección `[global]`. Sus alternativas son las siguientes:

```
security = share | user | server | domain
```

Desde la perspectiva del cliente, el nivel `share` corresponde al modo de seguridad

share y los niveles `user`, `server` y `domain` corresponden todos ellos al modo de seguridad `user`. A continuación se describen someramente los cuatro niveles.

El nivel `share` es utilizado normalmente en entornos en los cuales no existe un dominio Windows NT o 2000. En este caso, se asocia una contraseña por cada recurso, que debe proporcionarse correctamente desde el cliente cuando se pide la conexión.

En el nivel `user`, el encargado de validar al usuario es el sistema Unix donde Samba se ejecuta. La validación es idéntica a la que se realizaría si el usuario iniciase una sesión local en el ordenador Unix. Para que este método sea aplicable, es necesario que existan los mismos usuarios y con idénticas contraseñas en los sistemas Windows y en el sistema Unix donde Samba se ejecuta.

Recientemente, la utilización de este nivel se ha vuelto complicada, ya que en Windows 98, Windows NT (Service Pack 3 y posteriores) y Windows 2000, las contraseñas de los usuarios se transmiten cifradas por la red. Puesto que Samba no posee acceso a las contraseñas cifradas por Windows, el sistema Unix ya no puede realizar la validación. Existen dos métodos para resolver este problema. El primero consiste en modificar el registro del sistema Windows para permitir la transferencia de contraseñas sin cifrar por la red. El segundo método obliga a utilizar una tabla de contraseñas adicional en el sistema Unix, en la cual se almacenan las contraseñas cifradas de los usuarios Windows.

En el nivel `server`, Samba delega la validación del usuario en otro ordenador, normalmente un sistema Windows 2000. Cuando un cliente intenta iniciar una sesión con Samba, éste último intenta iniciar una sesión en el ordenador en el cual ha delegado la validación con la misma acreditación (usuario+contraseña) recibidos del cliente. Si la sesión realizada por Samba es satisfactoria, entonces la solicitud del cliente es aceptada. Este método aporta la ventaja de no necesitar que las contraseñas se mantengan sincronizadas entre los sistemas Windows y Unix, ya que la contraseña Unix no es utilizada en el proceso de validación. Adicionalmente, no hay inconveniente en utilizar contraseñas cifradas, ya que la validación la realiza un sistema W2000.

Por último, existe la posibilidad de utilizar el nivel `domain`. Este nivel es similar al nivel `server`, aunque en este caso el ordenador en el que se delega la validación debe ser un DC, o una lista de DCs. La ventaja de este método estriba en que el ordenador Samba pasa a ser un verdadero miembro del dominio W2000, lo que implica, por ejemplo, que puedan utilizarse las relaciones de confianza en las que participa el dominio W2000. Esto significa, en pocas palabras, que usuarios pertenecientes a otros dominios en los que los DCs confían son conocidos por Samba.

Dadas las ventajas del nivel `domain`, este documento se centra fundamentalmente en este método. Para detalles específicos de los otros niveles, se recomienda la consulta de la documentación original de Samba.

## 9.4. Configuración de Samba con el Nivel de Seguridad domain

Los pasos a seguir para configurar Samba con el nivel de seguridad domain son los siguientes:

1. Desde alguno de los DCs del dominio, dar de alta el sistema Unix donde se ejecuta Samba en el dominio<sup>2</sup>.
2. Detener el servidor Samba:

```
/etc/rc.d/init.d/smb stop
```

3. Utilizando `swat`, configurar el nivel de seguridad en el fichero `/etc/smb.conf`. En la sección `[global]` incorporar:

```
security = domain
workgroup = DOMINIO
encrypt passwords = yes
password server = DC1_DEL_DOMINIO, DC2_DEL_DOMINIO, ...
```

En esta configuración, la última línea hace referencia a los ordenadores que realizarán la autenticación de los usuarios. Esta línea se puede simplificar, y sustituir por: `password server = *`

4. Agregar el sistema Unix al dominio:

```
smbpasswd -j DOMINIO -r DC_DEL_DOMINIO
```

5. Iniciar el servidor Samba:

```
/etc/rc.d/init.d/smb start
```

## 9.5. Tratamiento de los Accesos como Invitado

Cuando se utiliza el nivel de seguridad domain, el tratamiento de los accesos como usuario invitado requiere algunas consideraciones. En particular, hay que tener en cuenta tres factores:

1. Si el usuario que accede ha sido acreditado por el dominio Windows NT o 2000 al cual pertenece el servidor Samba.
2. Si el usuario que accede existe, como usuario UNIX, en el servidor Samba.
3. El valor que tenga actualmente asignado la opción Samba `map to guest`.

La Tabla 9.2 a continuación indica, en función de los tres factores anteriores, si Samba permite o no el acceso y, en caso de permitirlo, si al usuario que accede se le considera como él mismo o como invitado.

<sup>2</sup>Si en dicho dominio ya existía una cuenta de máquina con el mismo nombre, hay que borrar dicha cuenta y volver a crearla.

MAP TO GUEST = NEVER (DEFECTO)		
En Samba...	En Windows...	
	Acreditado	No acreditado
Existe	Mismo usuario	No permitido
No existe	Invitado	No permitido
MAP TO GUEST = BAD USER		
En Samba...	En Windows...	
	Acreditado	No acreditado
Existe	Mismo usuario	No permitido
No existe	Invitado	Invitado
MAP TO GUEST = BAD PASSWORD		
En Samba...	En Windows...	
	Acreditado	No acreditado
Existe	Mismo usuario	Invitado
No existe	Invitado	Invitado

**Cuadro 9.2:** Distintos comportamientos del acceso a un servidor Samba.

En cualquier caso, para que el usuario “invitado” pueda acceder a un recurso compartido, este acceso tiene que permitirse expresamente para el recurso con la opción `guest ok` (cuyo valor por defecto es `no`).

## 9.6. El Sistema de Ficheros SMB para Linux

Linux dispone de soporte para el sistema de ficheros SMB. De esta forma, Linux, al igual que puede montar un directorio exportado vía NFS en un directorio local, puede montar un recurso SMB ofrecido por un servidor SMB (un sistema Windows o un servidor Samba, por ejemplo).

No obstante, existe una diferencia significativa entre NFS y SMB. En NFS no se requiere autenticar al usuario que realiza la conexión; el servidor NFS utiliza el UID del usuario del ordenador cliente para acceder a los ficheros y directorios exportados. Un servidor SMB, por contra, requiere autenticar al usuario, para lo que necesita un nombre de usuario y una contraseña. Por ello, para montar un recurso SMB se utiliza el mandato `mount` indicándole un tipo de sistema de archivos específico, denominado `smbfs`:

```
mount -t smbfs -o username=NOMBRE_DE_USUARIO,passwd=CONTRASEÑA,workgroup=
GRUPO o DOMINIO //ORDENADOR/RECURSO DIRECTORIO_LOCAL
```

Opcion	Significado	Valor por defecto
<code>security</code>	share user server domain	user
<code>workgroup</code>	dominio o grupo de trabajo	nulo
<code>encrypt passwords</code>	utilizar contraseñas cifradas de Windows	no
<code>password server</code>	lista de controladores de dominio para el nivel domain. Servidor de contraseñas para el nivel server	nulo
<code>netbios name</code>	Nombre NetBIOS del ordenador	Primer componente de su nombre DNS
<code>log level</code>	Detalle en la auditoría de Samba. Es un número que indica la cantidad de información a auditar. A mayor valor, más cantidad de información.	Se establece en el script que inicia el servicio Samba.
<code>log file</code>	Nombre del fichero donde se almacenan mensajes de auditoría de Samba.	Se establece en el script que inicia el servicio Samba.

**Cuadro 9.3:** Principales opciones de la sección `[global]` de Samba.

Si se omite la opción `passwd`, el sistema solicita al usuario que introduzca una contraseña. Si el servidor SMB valida al usuario, a partir del directorio `directorio_local` se consigue el acceso al recurso `//ordenador/recurso`.

## 9.7. Opciones del Servidor Samba

En la Tabla 9.3 se describen algunas opciones del servidor Samba. Estas opciones tan sólo son aplicables a la sección `[global]`.

## 9.8. Opciones del Recurso

En la Tabla 9.4 al final del capítulo se describen algunas opciones aplicables a cada recurso compartido. Pueden establecerse también en la sección `global`, siendo en este caso utilizadas como valores por defecto para cada recurso compartido.

Opcion	Significado	Valor por defecto
read only {yes/no}	Recurso de sólo lectura.	yes
browseable {yes/no}	El servicio aparece en la lista de recursos compartidos.	yes
path	Directorio asociado al servicio.	—
comment	Descripción del servicio.	—
guest ok {yes/no}	Permitir los accesos en modo invitado.	no
guest account	Si un acceso se realiza como invitado (usuario no conocido) se utiliza el usuario indicado para representar la conexión.	nobody
guest only	Cualquier acceso se realiza en modo invitado.	no
copy	Duplica un servicio ya declarado	—
force user	Los accesos al recurso se realizan como si el usuario que accede es el usuario indicado.	Se utiliza el mismo usuario que ha realizado la conexión.
force group	Los accesos al recurso se realizan como si el usuario que accede pertenece al grupo indicado.	Se utiliza el grupo primario del usuario que ha realizado la conexión.
hosts allow	Lista ordenadores a los que se permite acceder.	lista vacía (i.e., todos los ordenadores).
hosts deny	Lista ordenadores a los que no se les permite acceder. En caso de conflicto, prevalece lo indicado en <code>hosts allow</code> .	ningún ordenador.
valid users	Lista de usuarios que pueden acceder a este recurso.	lista vacía (i.e., todos los usuarios).
follow symlinks {yes/no}	Permitir el seguimiento de los enlaces simbólicos.	yes

**Cuadro 9.4:** Principales opciones de recurso de Samba.

# Apéndice A

## Ejemplo de Configuración DNS

### Índice General

---

A.1. Fichero /etc/named.conf . . . . .	109
A.2. Fichero /var/named/db.punto . . . . .	110
A.3. Fichero /var/named/db.in-addr.arpa . . . . .	111
A.4. Fichero /var/named/db.admon.com . . . . .	112
A.5. Fichero /var/named/db.42.158 . . . . .	113
A.6. Fichero /var/named/named.local . . . . .	115

---





## A.1. Fichero /etc/named.conf

```
// Configuración del servidor DNS adlin
//

options {
    directory "/var/named";
};

// Adlin es el servidor de toda la jerarquía DNS
// por lo que se declara maestro de la zona raíz.
//
zone "." {
    type master;
    file "db.punto";
};

// Un servidor "normal" nunca es maestro de la zona raíz,
// tan solo lo es de las zonas que tiene delegadas.
// No obstante, todos los servidores DNS necesitan conocer a
// los servidores DNS de la zona raíz para poder iniciar búsquedas
// en el espacio de nombres de dominio. Para ello, utilizan
// una zona tipo "hint", tal como se describe a continuación:
//
//
// zone "." {
//     type hint;
//     file "named.ca";
// };

// Adlin también se encarga de toda la zona de resolución
// inversa in-addr.arpa.
// Tampoco esta es una configuración normal.
//
zone "in-addr.arpa" {
    type master;
    file "db.in-addr.arpa";
};

// Ahora configuramos el dominio admon.com y su red asociada.
// Esta configuración es idéntica a la que se realizaría en
// un servidor de nombres en el cual se ha delegado la
// gestión de estas zonas.

// Maestro de la zona admon.com.
//
zone "admon.com" {
    type master;
    file "db.admon.com";
};
```

```
};

// Maestro de la zona de resolucion inversa
// de la red 158.42
//
zone "42.158.in-addr.arpa" {
type master;
file "db.42.158";
};

// Adicionalmente, si adlin fuese servidor esclavo de una zona
// se configuraría como sigue:
//
//
// zone "otrazona.com" {
//   type slave;
//   file "db.otrazona.com";
//   masters {
//     158.42.1.5; // El servidor maestro de otrazona.com
//   };
// };
//
// En este caso el fichero db.otrazona.com es donde
// se guarda la zona que transfiere el servidor maestro

// Maestro de la zona de resolucion inversa local
//
// En esta zona se declara la dirección local IP 127.0.0.1.
// Debe configurarse en todos los servidores de nombres.
//
zone "0.0.127.in-addr.arpa" {
type master;
file "named.local";
};
```

## A.2. Fichero /var/named/db.punto

```
;; Zona "."
;; La raiz de la jerarquia DNS
;;
;; Este es un caso especial, ya que adlin se declara como
;; el maestro de la zona raíz.
;;
;; Para un servidor normal, esta zona es generalmente de
```

```

;; tipo "hint" (no "master"), no se declara en ella
;; el registro de tipo SOA y la lista de servidores de nombres
;; enumera a los verdaderos servidores de la zona raíz.

$TTL 1m

. IN SOA      adlin.central.admon.com. namedmaster.central.admon.com. (
1  ; serial
12h ; refresh
30m ; retry
48h ; expire
1m  ; ttl
)

. IN NS adlin.central.admon.com.
adlin.central.admon.com. IN A 158.42.179.10

;; Estos son los servidores de verdad.

;; .                3600000      NS      A.ROOT-SERVERS.NET.
;; A.ROOT-SERVERS.NET. 3600000      A      198.41.0.4
;; .                3600000      NS      B.ROOT-SERVERS.NET.
;; B.ROOT-SERVERS.NET. 3600000      A      128.9.0.107
;; .                3600000      NS      C.ROOT-SERVERS.NET.
;; C.ROOT-SERVERS.NET. 3600000      A      192.33.4.12
;; .                3600000      NS      D.ROOT-SERVERS.NET.
;; D.ROOT-SERVERS.NET. 3600000      A      128.8.10.90
;; .                3600000      NS      E.ROOT-SERVERS.NET.
;; E.ROOT-SERVERS.NET. 3600000      A      192.203.230.10
;; .                3600000      NS      F.ROOT-SERVERS.NET.
;; F.ROOT-SERVERS.NET. 3600000      A      192.5.5.241
;; .                3600000      NS      G.ROOT-SERVERS.NET.
;; G.ROOT-SERVERS.NET. 3600000      A      192.112.36.4
;; .                3600000      NS      H.ROOT-SERVERS.NET.
;; H.ROOT-SERVERS.NET. 3600000      A      128.63.2.53
;; .                3600000      NS      I.ROOT-SERVERS.NET.
;; I.ROOT-SERVERS.NET. 3600000      A      192.36.148.17
;; .                3600000      NS      J.ROOT-SERVERS.NET.
;; J.ROOT-SERVERS.NET. 3600000      A      198.41.0.10
;; .                3600000      NS      K.ROOT-SERVERS.NET.
;; K.ROOT-SERVERS.NET. 3600000      A      193.0.14.129
;; .                3600000      NS      L.ROOT-SERVERS.NET.
;; L.ROOT-SERVERS.NET. 3600000      A      198.32.64.12
;; .                3600000      NS      M.ROOT-SERVERS.NET.
;; M.ROOT-SERVERS.NET. 3600000      A      202.12.27.33

```

### A.3. Fichero /var/named/db.in-addr.arpa

```

;; Zona de resolucion inversa para todas las redes

$TTL 1m

```

```
@ IN SOA adlin.central.admon.com. namedmaster.central.admon.com. (  
1 ; serial  
12h ; refresh  
30m ; retry  
48h ; expire  
1m ; ttl  
)  
  
@ IN NS adlin.central.admon.com.
```

## A.4. Fichero /var/named/db.admon.com

```
;; Zona admon.com.  
  
;; En este fichero @ representa a admon.com.  
;; Los nombres de dominio que no terminan con "." son relativos  
;; al dominio admon.com.  
;;  
  
;; Tiempo de vida por defecto para los registros de esta zona  
$TTL 1m  
  
;; INICIO DE AUTORIDAD DE ESTA ZONA  
  
@ IN SOA      adlin.central.admon.com. namedmaster.central.admon.com. (  
1 ; serial  
12h ; refresh  
30m ; retry  
48h ; expire  
1m ; ttl  
)  
  
;; SERVIDORES DE NOMBRES DE ESTA ZONA  
  
@ IN NS adlin.central.admon.com.  
  
;; RESOLUCIÓN DIRECTA PARA LOS ORDENADORES DE ESTA ZONA  
  
adlin.central IN A 158.42.179.10  
adwin.central IN A 158.42.179.14
```

```
;; ZONAS DELEGADAS

;; Delegación alacant:

alacant IN NS pc02.alacant.admon.com.
alacant IN NS pc03.alacant.admon.com.
pc02.alacant IN A 158.42.201.2
pc03.alacant IN A 158.42.201.3

;; Delegación alcoi:

alcoi IN NS pc06.alcoi.admon.com.
alcoi IN NS pc07.alcoi.admon.com.
pc06.alcoi IN A 158.42.202.6
pc07.alcoi IN A 158.42.202.7

;; Delegación castello:

castello IN NS pc10.castello.admon.com.
castello IN NS pc11.castello.admon.com.
pc10.castello IN A 158.42.203.10
pc11.castello IN A 158.42.203.11

;; Delegación gandia:

gandia IN NS pc14.gandia.admon.com.
gandia IN NS pc15.gandia.admon.com.
pc14.gandia IN A 158.42.204.14
pc15.gandia IN A 158.42.204.15

;; Delegación valencia:

valencia IN NS pc18.valencia.admon.com.
valencia IN NS pc19.valencia.admon.com.
pc18.valencia IN A 158.42.205.18
pc19.valencia IN A 158.42.205.19
```

## A.5. Fichero /var/named/db.42.158

```
;; Zona de resolucioin inversa de la red 158.42/16
;;
;; En este fichero @ representa a 42.158.in-addr.arpa.
```

```
;; Los nombres de dominio que no terminan con "." son relativos
;; al dominio 42.158.in-addr.arpa.
;;

;; Tiempo de vida por defecto para los registros de esta zona

$TTL 1m

;; INICIO DE AUTORIDAD DE ESTA ZONA

@ IN SOA adlin.central.admon.com. namedmaster.central.admon.com (
1 ; serial
12h ; refresh
30m ; retry
48h ; expire
1m ; ttl
)

;; SERVIDORES DE NOMBRES DE ESTA ZONA

@ IN NS adlin.central.admon.com.

;; RESOLUCIÓN INVERSA DE LOS ORDENADORES DE ESTA ZONA

10.179 IN PTR adlin.central.admon.com.
14.179 IN PTR adwin.central.admon.com.

;; ZONAS DELEGADAS

;; Delegación alacant:

201 IN NS pc02.alacant.admon.com.
201 IN NS pc03.alacant.admon.com.
pc02.alacant IN A 158.42.201.2
pc03.alacant IN A 158.42.201.3

;; Delegación alcoi:

202 IN NS pc06.alcoi.admon.com.
202 IN NS pc07.alcoi.admon.com.
pc06.alcoi IN A 158.42.202.6
pc07.alcoi IN A 158.42.202.7
```

```
;; Delegación castello:

203 IN NS pc10.castello.admon.com.
203 IN NS pc11.castello.admon.com.
pc10.castello IN A 158.42.203.10
pc11.castello IN A 158.42.203.11
```

```
;; Delegación gandia:

204 IN NS pc14.gandia.admon.com.
204 IN NS pc15.gandia.admon.com.
pc14.gandia IN A 158.42.204.14
pc15.gandia IN A 158.42.204.15
```

```
;; Delegación valencia:

205 IN NS pc18.valencia.admon.com.
205 IN NS pc19.valencia.admon.com.
pc18.valencia IN A 158.42.205.18
pc19.valencia IN A 158.42.205.19
```

## A.6. Fichero /var/named/named.local

```
;; Zona de resolucio local
;;

$TTL 1m

@ IN SOA localhost. root.localhost. (
1 ; serial
12h ; refresh
30m ; retry
48h ; expire
1m ; ttl
)

@ IN NS localhost.
1 IN PTR localhost.
```

