# 3. Gestión y administración de usuarios. Sistemas de autentificación

- Cuentas de usuario. Creación, eliminación, cambio de propiedades de usuarios locales.
- Autentificación local de usuarios. Grupos locales. Permisos de acceso a recursos.

# Introducción

- Un usuario es una persona que trabaja en el sistema
- Un pseudo-usuario es una entidad que puede ejecutar programas y poseer ficheros
- Un usuario se caracteriza por:
  - Un nombre de usuario (logname o username)
  - Un Identificador de Usuario (UID). El sistema identifica internamente al usuario por un número, y no por su nombre
  - Un conjunto de grupos al que pertenece (GID, cada grupo tiene asociado otro número)

# Introducción

- Los ficheros de configuración de los usuarios son:
  - /etc/passwd: Identifica las cuentas de los usuarios
  - /etc/shadow: Password encriptados
  - /etc/group: Grupos y usuarios miembros

# Añadir un usuario al sistema

- La herramienta **adduser NombreDeUsuario** crea el usuario cuyo nombre se le indica. Al nuevo usuario se le asigna el primer UID libre (el primero es 500) y crea el directorio del usuario. A continuación crea los ficheros de inicialización (.bash_profile, .bashrc, etc.)

- A continuación, se ejecuta **passwd NombreDeUsuario**, con lo que se actualiza el contenido de /etc/shadow

- La orden **userdel** elimina una cuenta (no borra los datos del directorio)

- Hay herramientas gráficas (system-config-users)

# Fichero /etc/passwd

## Fichero /etc/passwd

- Contiene la lista de usuarios definidos en el sistema
- Formato: *nombre:password:uid:gid:gecos:home:shell*
  - nombre ⇒ Nombre del usuario, **logname** o **username**
  - password ⇒ contraseña cifrada o :
    - «*» o «!!» ⇒ la cuenta está desactivada o bloqueada
    - «x» ⇒ Las *shadow* están activas, la contraseña cifrada se guarda en **/etc/shadow**
  - uid ⇒ identificador del usuario
  - gid ⇒ identificador del grupo primario al que pertenece
  - gecos ⇒ campo de información referente al usuario (nombre, teléfono, . . . )
  - home ⇒ Path del directorio «HOME» del usuario
  - shell ⇒ Intérprete de órdenes que se ejecutará al entrar al sistema
- **/usr/sbin/vipw** ⇒ Para modificar el fichero manualmente
- El usuario propietario es el **root** y el grupo **root**

# Fichero shadow

- /etc/shadow es un fichero que está protegido contra lectura por todos los usuarios excepto root (/etc/passwd puede verse por cualquier usuario)
- Contiene las claves encriptadas (en /etc/passwd aparece una “x”, indicando que la clave está en el fichero shadow)
- Las cuentas pueden estar sometidas a restricciones de tiempo respecto a su validez o a su contraseña. Estas opciones se almacenan en el fichero /etc/shadow. Entre otras, son:
  - Fecha del último cambio de password
  - Mínimo número de días antes de cambiar la contraseña
  - Máximo número de días sin cambiar contraseña
  - Días antes de que expire la contraseña en que se avisa al usuario
  - Fecha de expiración de la cuenta

# /etc/login.defs y /etc/skel

- El fichero /etc/login.defs contiene los valores por defecto de la antigüedad de una cuenta nueva

- La orden **chage** permite cambiar estos valores a cada usuario. También se puede hacer con las opciones de la orden passwd.

- En /etc/skel están los patrones de los ficheros de inicialización de cada usuario:

| | |
|---|---|
| Se ejecuta al hacer un login (PATH, variables de entorno, umask, funciones de inicialización, etc.) | *.bash_profile* en Bourne Again Shell (bash)<br>*.profile* en Bourne Shell (sh)<br>*.login* en C Shell (csh) |
| Cada vez que se ejecuta un shell (alias, var. del propio shell, etc.) | *.bashrc* en Bourne Again Shell (bash)<br>*.cshrc* en C Shell (csh) |
| Al salir el usuario | *.bash_logout* en Bourne Again Shell (bash)<br>*.logout* en C Shell (csh) |

# Shell por defecto

- El último campo de /etc/passwd contiene el intérprete de órdenes que se ejecuta al entrar al sistema

- En /etc/shells se indican los shells permitidos. Sólo afecta a las cuentas de nueva creación. Si se elimina una línea de este fichero los usuarios que estén usando ese shell podrán seguir haciéndolo.

- Con la orden **chsh** un usuario puede cambiar su shell

- Si se desea que un usuario no entre en sesión se le puede poner como shell “/bin/false” o “/bin/nologin”

- También se puede reemplazar el shell por un fichero ejecutable, que se lanzará cuando el usuario entra en sesión.

# Cuentas restringidas

- Pueden hacerse cuentas en que los usuarios tengan restringidas algunas acciones. Esto se puede conseguir haciendo que el shell sea un fichero ejecutable que realiza una tarea determinada (por ejemplo, copias de seguridad, apagado de la máquina, etc.)

# Grupos

- Un grupo es una colección de usuarios que comparten recursos o ficheros del sistema
- Se les puede asignar permisos a un conjunto de usuarios
- El fichero de configuración es el /etc/group, con formato nombre:*:gid:lista de usuarios (separados por comas)
- Un grupo puede estar definido de forma implícita (p.e., no aparece en /etc/group pero sí en /etc/passwd). Un usuario puede pertenecer a más de un grupo. El grupo primario es el de /etc/passwd. Los otros grupos son secundarios.
- Al crear un fichero, el grupo activo es el grupo del usuario, que es el primario salvo que se haya modificado el grupo activo con **newgrp.** Para acceder a un archivo se usan todos los grupos del usuario
- La orden **groups** lista todos los grupos a los que pertenece un usuario.

# Usuarios estándar

- root: Administrador
- bin, daemon, lp, sync, shutdown: Para poseer ficheros o ejecutar servicios (pseudo usuarios)
- mail, news, ftp: Asociadas a ciertas herramientas
- postgres, mysql, xfs: Creados por ciertas herramientas para ejecutar sus servicios
- nobody (nfsnobody): Usada por NFS

# Grupos estándar

- root, sys
- bin, daemon, adm, lp, disk, mail, ftp, nobody, etc.
- kmem: Grupo propietario de programas para leer la memoria del kernel
- Cada nuevo usuario es asignado a un grupo en el que él es el único miembro, y cuyo nombre es igual que el nombre de usuario.

# Resumen (1)

- Cada usuario tiene una cuenta, que tiene asociados un conjunto de privilegios, y un espacio en disco.

- Hay dos tipos de cuentas: de sistema (como root o apache) y de usuario. Las cuentas de sistema se crean durante la instalación por diferentes demonios y utilidades para realizar tareas de sistema. Los UIDs entre 0 y 499 se reservan para cuentas de sistema. Las cuentas de usuario comienzan en 500.

- UID significa User Identifier. Es un equivalente numérico del nombre de usuario. GID significa Group Identifier. El UID 0 se reserva para root.

# Resumen (2)

- Cuando se ejecuta la orden

  **/usr/sbin/useradd dan**

ocurren los siguientes pasos:

1. Se añade a /etc/passwd una linea similar a

   **dan:x:502:502::/home/dan:/bin/bash**

donde

- **dan** es el nombre de usuario
- **x** es el campo de password; x significa que el campo está vacío y que una password encriptada se incluye en /etc/shadow
- el primer **502** es el uid
- el segundo **502** es ell gid
- campo vacío (suele contener el nombre completo del usuario)
- **/home/dan** es el directorio home del usuario
- **/bin/bash** es el shell por defecto

Se usan shadow passwords por defecto, los directorios de usuario son subdirectorios de /home, el shell por defecto es bash y se crea un nuevo grupo con el mismo nombre que el usuario

# Resumen (3)

2. Se le añade una línea al fichero /etc/shadow, similar a:

**dan:!!:13490:0:99999:7:::**

con 8 campos que significan:

1. **dan** es el nombre de usuario
2. **!!** indica que la password aún no ha sido puesta y que la cuenta está bloqueada
3. **13490** es el número de días (desde 1 de Enero de 1970) que hace que se ha cambiado la password
4. **0** representa el número de días deben pasar antes de que la clave pueda cambiarse
5. **99999** es el número de días que deben pasar antes de que la clave deba cambiarse obligatoriamente
6. **7** es el número de días que pasarán antes de que la password expire
7. **Primer campo vacío**: número de días después de la expiración de la password antes de que se deshabilite la cuenta
8. **Segundo campo vacío**: Número de días desde 1/1/1970 que la cuenta ha sido deshabilitada

**El comando useradd no crea una password**

# Resumen (4)

4. Se añade una línea a **/etc/group**, similar a

**dan:x:502**

cuyos campos significan:

    1. **dan**: es el nombre del grupo
    2. **x**: es el campo de la password de grupo (x = shadow)
    3. **502**: es el gid

5. Se añade una línea a **/etc/gshadow**, similar a

**dan:!::**

cuyos campos significan:

    1. **dan**: nombre de grupo
    2. **!** es la password de grupo (bloqueada)

6. Se crea el directorio de usuario **/home/dan**, propiedad del usuario dan y grupo dan, permiso de lectura, escritura y ejecución
7. Se copian los ficheros del directorio **/etc/skel** (por ejemplo, .bashrc y .bash_profile)
8. El administrador del sistema debe correr el comando **/usr/bin/passwd** para indicar la password del usuario y desbloquear la cuenta

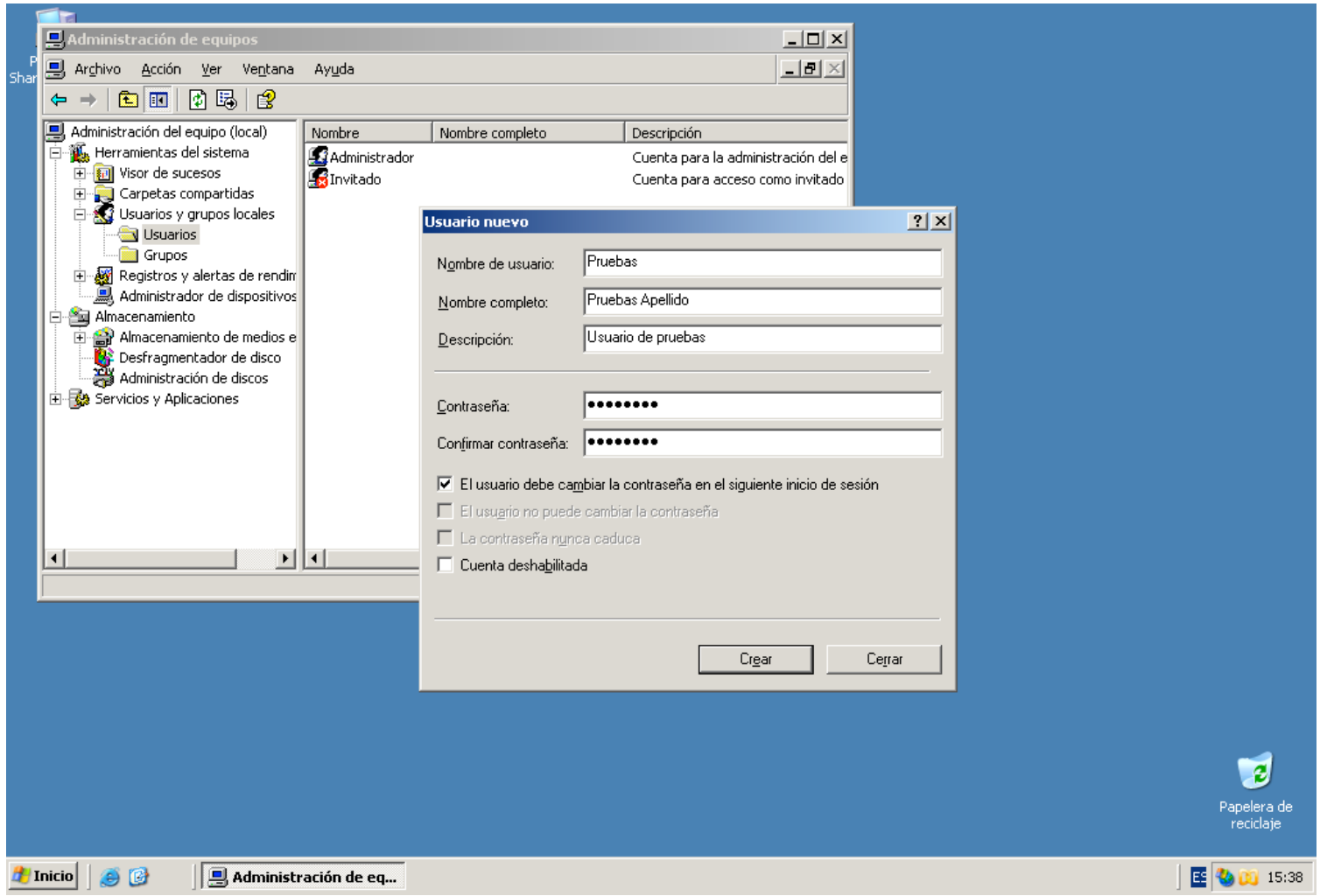
# Usuarios y grupos locales Windows 2008

- Las cuentas locales están diseñadas para el personal de apoyo al administrador (las de Dominio tienen otra función, como se verá más adelante).

- Al instalar Windows se crea automáticamente la cuenta de administrador. Esta es una cuenta local, y si se instala Active Directory tiene también privilegios de acceso en el dominio

- Pueden crearse cuentas locales adicionales. Por defecto, se crean Invitado y Soporte, ambas deshabilitadas

# Herramientas para creación de usuarios locales



# Atributos de protección de los procesos Unix

- **Identificadores del usuario propietario del proceso:** cada proceso contiene dos versiones del identificador de usuario, la denominada versión real (rUID) y la versión efectiva (eUID). La versión real se corresponde con el usuario que creó el proceso. La versión efectiva se corresponde con el usuario bajo el cual se comporta el proceso y es la utilizada en el mecanismo de protección. Ambos identificadores suelen ser iguales, salvo que intervenga el bit SETUID que se menciona más adelante.

- **Identificadores del grupo propietario del proceso**: versiones reales (rGID) y efectiva (eGID) del identificador. La versión real es el grupo primario al que pertenece el usuario que creó el proceso. La efectiva se corresponde con el grupo bajo el que se comporta el proceso, y coincide con la real salvo que intervenga el bit SETGID.

- **Lista de grupos suplementarios**: Grupos suplementarios del usuario que creó el proceso

# Atributos de protección de los procesos Unix

- Los atributos de protección de los ficheros se asignan al proceso en el momento de su creación, y son heredados de su proceso padre.

- Cada usuario conectado a un sistema Unix posee un proceso dentro del entorno de ventanas o el intérprete de órdenes. Este proceso es el padre de todos los procesos que genera el usuario. Este proceso shell no recibe sus atributos por herencia; en el momento que un usuario se acredita ante el sistema con su nombre de usuario y contraseña asociadas, se extraen los atributos rUID y gUID de /etc/passwd. La lista de grupos suplementarios se confecciona a partir de /etc/group. Los atributos eUID y eGID se asignan a los valores respectivos de rUID y gUID.

- El comando **id** muestra todos estos atributos

# Atributos de protección de los ficheros Unix

- Los atributos de un fichero que intervienen en el mecanismo de protección son:
  - **OwnerUID:** Identificador del usuario propietario del fichero
  - **OwnerGID:** Identificador del grupo propietario del fichero
  - **Bits de permiso:** Un total de 12 bits que expresan las propiedades que son permitidas en función del proceso que acceda a este fichero

# Significado de los bits de permiso en Unix

| 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. |
| 4 | Escritura para el grupo. |
| 3 | Ejecución para el grupo. |
| 2 | Lectura para el resto de usuarios. |
| 1 | Escritura para el resto de usuarios. |
| 0 | Ejecución para el resto de usuarios. |

# Significado de los bits de permiso en Unix

- El significado de los bits de lectura, escritura y ejecución es diferente en función del tipo de archivo que los defina.
    - Para *ficheros regulares*, permiten leer, modificar y ejecutar el fichero
    - En un *directorio*,
        - Lectura: permite listar el contenido del directorio
        - Escritura: permite crear, eliminar o renombrar ficheros dentro del directorio al que se aplica el bit
        - Ejecución: permite la utilización del directorio al que se aplica este bit para formar parte de un nombre de ruta

# Significado de los bits de permiso en Unix

- No existe un bit de permiso específico para el borrado de un fichero/directorio, sino que ese permiso se controla desde el directorio donde reside el fichero.

- El bit *sticky* se emplea 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.

- El atributo ownerUID se modifica con el comando **chown**

- El atributo ownerGID se modifica con el comando **chgrp**

- Los bits de permiso se modifican con **chmod**

# Permisos de acceso a recursos locales Unix

- Hay tres permisos diferentes para ficheros, directorios y aplicaciones. Estos permisos se indican con los símbolos **r**(lectura), **w**(escritura), **x**(ejecución), **-**(sin acceso)
- Cada uno de los tres permisos puede asociarse a tres diferentes categorías: **owner, group, everyone**
- Los permisos se muestran con ls -l:

  **-rwxrwxr-x** 1 juan gjuan 0 Sep 26 12:25 nombrefichero
- Hay tres grupos de tres letras: rwx (owner - u), rwx (group -g ), r-x (everyone - a). En este ejemplo, sólo pueden modificar el fichero ejecutable llamado “nombrefichero” los miembros del grupo “gjuan” o su propietario “juan”.

# Reglas de protección básicas Unix

- Las reglas de protección básicas se activan cuando un proceso notifica al sistema que desea utilizar un determinado fichero. El proceso notifica al sistema si desea realizar una operación de lectura, escritura o ejecución.
  - Si el eUID del proceso es 0 se concede el permiso
  - Si el eUID del proceso coincide con ownerUID del fichero, se autoriza la operación si está permitida en los bits 6 a 8 (propietario)
  - Si el eGID del proceso o de alguno de los grupos suplementarios del proceso es igual al ownerGID del fichero, se autoriza la operación si está permitida en los bits de 3 a 5 (grupo)
  - Si no se cumple nada de lo anterior, se autoriza la operación si está permitida en los bits del 0 al 2

# Reglas de protección básicas Unix

- Sólo se aplica una regla, aquella que corresponde a la primera condición que se cumple para los atributos del proceso: 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 los tres bits correspondientes.

# Cambio de atributos de protección en ficheros

- Cambio en los bits de permiso: un proceso sólo puede cambiar los bits de permiso de un fichero si el eUID del proceso es 0 o el eUID del proceso es igual al ownerUID del fichero
- Cambio de propietario: el cambio de propietario de un fichero sólo puede realizarlo el superusuario
- Cambio de grupo propietario: el superusuario o bien el propietario del fichero siempre que el nuevo ownerGID del fichero sea uno de los grupos de usuario

# SETUID y SETGID en 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
  - Si el fichero ejecutable tiene el bit SETUID activo, el eUID del proceso que ejecita 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

# SETGID en directorios

- Este bit sirve para facilitar el trabajo en grupo cuando varios usuarios deben acceder a una colección común de ficheros y directorios. Si un directorio tiene el bit SETGID activo, entonces
  - Si se crea un fichero en el directorio, el ownerGID del fichero es hecho igual al ownerGID del directorio (y no igual al eGID del proceso creador)
  - Si se crea un subdirectorio, su ownerGID es hecho igual al ownerGID del directorio y su bit SETGID es activado

# La máscara de creación de ficheros

- Las utilidades de creación de ficheros utilizan por defecto la palabra rw-rw-rw- en ficheros no ejecutables, rwxrwxrwx si se crea un directorio o un ejecutable.

- Cada usuario puede notificar al sistema que desea cambiar esta máscara mediante la orden umask. Cada bit activo en la máscara se desactiva cuando se cree un fichero. Por ejemplo, la máscara 022 desactiva los bits de escritura para el grupo y el resto de usuarios.

# Resumen de los bits de permiso especial

- Hay tres permisos especiales: **setuid, setgid, sticky bit**
  - **setuid**: se usa en aplicaciones. La aplicación corre como el propietario del fichero y no como el usuario que la ejecuta. Se indica con una **s** en lugar de la **x** en la categoría “owner”. Si el propietario no tiene permiso de ejecución, con una **S** mayúscula.
  - **setgid**: la aplicación corre como el grupo del usuario que la lanza. Si se aplica a un directorio, los ficheros creados en el directorio son propiedad del grupo que posee el directorio, y no del grupo del usuario que crea el fichero. Se indica con una **s** en lugar de la **x** de la categoría “grupo”.
  - **sticky bit**: se usa en directorios. Este bit indica que un fichero creado en un directorio sólo puede ser eliminado por el usuario que lo creó. Se indica mediante el carácter **t** en vez de la **x** en la categoría everyone. Si la categoría “everyone” no tiene permiso de ejecución, la **T** es mayúscula.

# Modelo de protección Windows: derechos y permisos

- El modelo de protección de Windows establece la forma en que el sistema lleva a cabo el control de acceso de cada usuario y grupo de usuarios.

- En el caso del sistema y sus recursos, se definen dos conceptos distintos y complementarios: el concepto de derecho y el concepto de permiso, respectivamente.

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

# Atributos de protección de los procesos

- Cuando un usuario es autorizado a conectarse interactivamente a un sistema Windows, el sistema construye para él una acreditación denominada **Security Access Token o SAT**.

- Esta acreditación contiene la información de protección del usuario, y Windows la incluye en los procesos que crea para dicho usuario. De esta for- ma, 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.

# Atributos de protección del SAT

- **SID**: Identificador único de usuario
- **SID de sus grupos:** Lista de grupos a los que pertenece el usuario
- **Derechos:** 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.
- El nivel de acceso de un usuario incluye implícitamente los niveles de los grupos a los que pertenece.

# 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).

- 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.

- Windows distingue entre dos tipos de derechos: **los derechos de conexión (logon 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.

# Derechos de conexión

| 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. |
| Inicio de sesión local | Permite/impide al usuario iniciar una sesión local en el ordenador, desde el teclado del mismo. |

# Privilegios

| 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.). |

# Conflictos

- 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 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.

# Atributos de protección de los recursos

- En NTFS, cada carpeta o archivo posee los siguientes atributos de protección:
  - **SID del propietario:** Inicialmente, el propietario es siempre el usuario que ha creado el archivo o carpeta, aunque este atributo puede ser luego modificado.
  - **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. 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).
  - **Lista de control de acceso de seguridad**

# Atributos de protección de los recursos

- **SID del propietario:**
- **Lista de control de acceso de protección:**
- **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 (las entradas de este registro pueden consultarse con el Visor de Sucesos). 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.

# Lista de control de acceso de protección

- La lista de control de acceso de protección se divide 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. Cada entrada liga a un SID de usuario o grupo con la concesión o denegación de un permiso concreto (o conjunto de permisos). Los diferentes permisos que se pueden asignar a usuarios o grupos se explican más adelante.

- 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 de Windows: cada archivo o carpeta puede heredar implícitamente los permisos establecidos para la carpeta que lo contiene y puede además definir permisos propios (explícitos). 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.

# Reglas de asociación de permisos a recursos

- 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:
  - 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.
  - 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 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).

# Permisos estándar e individuales

- Windows distingue entre los permisos estándar e individuales de carpetas (directorios) y los de archivos. *Los permisos estándar son combinaciones predefinidas de permisos individuales*, que son aquellos que controlan cada una de las acciones individuales que se pueden realizar sobre carpetas y archivos.

- Cada permiso puede ser positivo o negativo, es decir, que realmente cada permiso permite o deniega la acción correspondiente. 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 excepto el de Listar (sólo definido para carpetas).

# Permisos estándar sobre carpetas

| 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 `Leer` y de `Listar`. |
| `Modificar` | Permite eliminar la carpeta más todos los permisos de `Escribir` y de `Leer y Ejecutar`. |
| `Control Total` | Permite cambiar permisos, tomar posesión y eliminar subcarpetas y archivos (aun no teniendo permisos sobre ellos), así como todos los permisos anteriores. |

# Permisos estándar sobre archivos

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

# Permisos individuales

- **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 co- mo oculto o sólo lectura, definidos en NTFS.
- **Leer atributos extendidos:** Permite ver los atributos extendidos del archivo o car- peta. (Estos atributos están definidos por los programas y pueden variar).
- **Crear ficheros/escribir datos:** Aplicado a una carpetas, permite crear archivo en ella. Aplicado a un archivo, permite modificar y sobreescribir su contenido.
- **Crear carpetas/anexar datos:** Aplicado a una carpeta, permite crear subcarpetas en ella. Aplicado a un archivo, permite añadir datos al mismo.
- **Escribir atributos:** Permite modificar los atributos de un archivo o carpeta
- **Escribir atributos extendidos:** Permite modificar los atributos extendidos de un archi- vo 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.

# Reglas de protección Windows

- 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 proceso 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 son acumulativos: un proceso de usuario posee implícitamente todos los permisos correspondientes a los SIDs de su acreditación, 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.

# Reglas de protección Windows

- El sistema explora secuencialmente las entradas de las DACLs de protección de un objeto hasta que se cumple alguna de las condiciones siguientes:
    - 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.
    - 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.
    - 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.
- El orden en que Windows 2000 establece las entradas de las DACLs de cada objeto es el siguiente: permisos negativos explícitos, permisos positivos explícitos, permisos negativos heredados y permisos positivos heredados.