

Programación y Administración de Sistemas

4. Gestión de usuarios

Pedro Antonio Gutiérrez

Asignatura "Programación y Administración de Sistemas"

2º Curso Grado en Ingeniería Informática

Escuela Politécnica Superior

(Universidad de Córdoba)

pagutierrez@uco.es

6 de marzo de 2017



Objetivos del aprendizaje I

- Definir qué son usuarios del sistema, las características de los mismos y sus ficheros de configuración.
- Enumerar y explicar los campos del fichero `/etc/passwd` y `/etc/shadow`.
- Explicar las características que deberían tener las contraseñas para los usuarios.
- Explicar el mecanismo de *shadow passwords* y el mecanismo de cifrado de contraseñas que evita guardar las contraseñas del sistema en texto plano.
- Enumerar los mecanismos de revocación de contraseñas, las restricciones de tiempo en cuanto a la validez de las contraseñas y las herramientas de administración que permiten configurarlas.
- Cambiar el intérprete de órdenes por defecto de los usuarios.
- Configurar cuentas restrictivas para usuarios especiales.

Objetivos del aprendizaje II

- Enumerar los pasos para añadir un usuario al sistema.
- Utilizar herramientas administrativas para añadir o modificar cuentas de usuario.
- Establecer el objetivo de los grupos de usuarios, identificar grupo primario y grupo activo de un usuario, enumerar y explicar los campos del fichero `/etc/group`.
- Configurar grupos con contraseñas.
- Utilizar las distintas herramientas administrativas para grupos.
- Identificar usuarios y grupos estándar en un sistema GNU/Linux.

4.1. Introducción.

- 4.1.1. Definición de usuario.
- 4.1.2. Características de un usuario.
- 4.1.3. Ficheros de configuración de usuarios.

4.2. Usuarios.

- 4.2.1. Fichero `/etc/passwd`, contraseñas y *shadow passwords*.
 - 4.2.1.1. Estructura del fichero `/etc/passwd`.
 - 4.2.1.2. Características deseables para las contraseñas.
 - 4.2.1.3. Sistema de protección *shadow passwords*: fichero `/etc/shadow`, algoritmos criptográficos de generación de resumen (*hash*).
- 4.2.2. Restricciones de tiempo.
- 4.2.3. Ficheros de inicialización.
- 4.2.4. Intérprete de órdenes por defecto y cuentas restrictivas.
- 4.2.5. Pasos para añadir nuevos usuarios al sistema.
- 4.2.6. Herramientas para crear/modificar cuentas de usuario.

4.3. Grupos.

- 4.3.1. Fichero de configuración `/etc/group`.

4.3.2. Grupos con contraseñas.

4.3.3. Herramientas de administración de grupos.

4.4. Usuarios y grupos estándar.

- Cuestionarios objetivos.
- Pruebas de respuesta libre.
- Tareas de administración.

Introducción

Definición de **usuario**

- Persona que trabaja en el sistema, editando ficheros, ejecutando programas...
- **Pseudo-usuario**: entidad, que sin ser una persona, puede ejecutar programas o poseer ficheros (se les reserva identificadores de 0 a 499).

Características de un usuario

- Nombre de usuario (logname o username).
- Identificador de usuario (UID): el sistema trabaja, internamente, con el UID y no con el nombre de usuario.
- Identificadores de los grupos a los que pertenece (GIDs).



Introducción

Ficheros de configuración:

- `/etc/passwd` ⇒ información de las cuentas de usuarios.
- `/etc/shadow` ⇒ *passwords* cifradas (*hash* de las contraseñas) e información de “envejecimiento” de las cuentas.
- `/etc/group` ⇒ definición de los grupos y usuarios miembros.
- `/etc/gshadow` ⇒ *passwords* de grupos cifradas.



Fichero /etc/passwd

- Contiene la lista de usuarios del sistema y sus contraseñas.
- 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.



Fichero `/etc/passwd`

- El propietario del fichero es **root** y el grupo **root**.
- Los permisos del fichero son **`rw-r--r--`**.
- El programa `/usr/sbin/vipw` permite editar el fichero manualmente.
- El programa `pwck` verifica la integridad de `/etc/passwd` y `/etc/shadow`.
- Se permite el acceso al fichero `/etc/passwd` en modo lectura para poder leer información del usuario, pero no se debería permitir acceso a las *passwords* (aunque estén cifradas).



Fichero /etc/passwd

```
tux:x:1001:100:The Linux penguin:/home/tux:/bin/bash
```

- Standard shell
- Home directory
- Comments field
- GID of primary group
- UID
- Password
- User name



Contraseñas

- `passwd <nombre_usuario>` \Rightarrow asignar contraseña a un usuario (o cambiarla).
- Elección de una contraseña adecuada:
 - **No utilizar:**
 - Tu nombre, parte de él, o el de alguien cercano a ti.
 - Números significativos para ti o alguien cercano.
 - Nombre, n^o, lugar o persona, relacionados con tu trabajo.
 - Palabras que estén en el diccionario.
 - Nombres de gente famosa, lugares, películas, publicidad...
 - **Consejos:**
 - Introducir 2 o más caracteres extras, símbolos especiales...
 - Escribir mal las palabras.
 - Utilizar mayúsculas y minúsculas, pero no de forma evidente.
 - Concatenar, embeber o mezclar 2 o más palabras.
 - Usar caracteres poco comunes: \$, &, #...



Contraseñas

- La contraseña **se debe** cambiar cuando:
 - Se sospecha que alguien la ha podido conocer o averiguar.
 - Se sospecha que alguien ha conseguido el fichero con las contraseñas (/etc/passwd o /etc/shadow).
 - Un usuario se marcha del trabajo ⇒ cambiar todas las que conozca.
 - Un administrador del sistema se va ⇒ cambiar TODAS.
 - Un intruso ha conseguido entrar en el sistema.
- **Periódicamente**, se debe forzar a que los usuarios cambien sus contraseñas, incluido el administrador.
 - Por otro lado, si se obliga a los usuarios a cambiar su contraseña con **demasiada frecuencia**, lo normal es que elijan malas contraseñas, fáciles de adivinar...



Shadow passwords

- Permiten que las contraseñas cifradas no se guarden en el fichero /etc/passwd sino en /etc/shadow (más restringido).
- /etc/shadow tiene los permisos **rw-----**, y el usuario y grupo propietario es **root**.
 - En las últimas versiones, tiene permisos **rw-r-----** y el grupo propietario es shadow (flexibilidad para comprobación de contraseña por usuarios que pertenezcan a dicho grupo).
- Este fichero guarda para cada usuario del sistema, la contraseña cifrada junto con su **información de envejecimiento**.
- Solo para aquellos usuarios que tengan una “x” en /etc/passwd.
- Por defecto, están activas y se actualizan automáticamente.



Shadow passwords

geeko:mostSt1zdI45I:12623:1:99999:14:-1:12134:

- Day on which account is locked (days since 1.1.1970)
- For how many days is password valid, although password has expired.
- How many days before password expires should user be warned?
- Days after which password must be changed
- Days after which password may be changed
- Date of last change (days since 1.1.1970)
- encrypted password
- User name



Shadow passwords

`nom:pass:changed:minlife:maxlife:warn:inactive:expired:unused`

- **nom** ⇒ nombre del usuario, *logname* o *username*.
- **pass** ⇒ contraseña cifrada.
 - `mkpasswd --method=sha-512` contraseña salt
- Comandos de actualización:
 - `pwconv` ⇒ crear y actualizar el fichero `/etc/shadow`.
 - `pwunconv` ⇒ desactivar los *shadow passwords*.



Shadow passwords

- Para cifrar una contraseña, se utilizan algoritmos criptográficos de generación de resumen (función *hash*, $H(\cdot)$).
 - 1 El mensaje en este caso es la contraseña (C).
 - 2 salt (S) es una palabra aleatoria que se concatena a los bytes de contraseña \rightarrow dificulta ataques con diccionarios y tablas de *hash* precomputadas; añade aleatoriedad al resumen.
 - 3 El sistema concatena C con S , $\{C, S\}$, calcula el resumen $F = H(\{C, S\})$ y almacena S y F .
 - 4 Cuando el usuario introduce una contraseña C' se repite todo el proceso: $F' = H(\{C', S\})$.
 - 5 Si $F = F'$, entonces el usuario puede entrar al sistema.



Shadow passwords

- Propiedades deseables de las funciones de resumen:
 - Dado C , debe ser fácil calcular $H(C) \rightarrow$ para que el coste computacional no sea alto.
 - Dado $H(C)$, debe ser extremadamente difícil calcular $C \rightarrow$ para que las contraseñas originales no se puedan conocer sabiendo el resumen (fugas de información).
 - Dado C , debe ser muy difícil encontrar otro mensaje C' tal que $H(C) = H(C') \rightarrow$ para que dos usuarios no terminen con la misma contraseña.
- Este tipo de funciones se denominan **funciones de dispersión de un solo sentido**.



Shadow passwords: Algoritmos de hash

- Dos algoritmos:
 - **MD5** (*Message-Digest algorithm 5*):
 - Aplica funciones no lineales a los 17 segmentos de 32 bits de un bloque de 512 bits.
 - Se obtiene un resumen de 128 bits.
 - Obtener suma MD5 (GNU/Linux):

```
md5sum Fichero.ext > Fichero.md5
```

- Chequear suma MD5 (GNU/Linux) (se busca un fichero con el nombre correcto en la carpeta actual):

```
md5sum -c Fichero.md5
```



Shadow passwords: Algoritmos de hash

- Dos algoritmos:
 - **SHA** (*Secure Hash Algorithm*):
 - Estándar del NIST.
 - Parecido a MD4, pero genera resúmenes más grandes, que lo hacen más seguro contra ataques de fuerza bruta o del cumpleaños.
 - Se pueden considerar 160, 224, 256, 384 o 512 bits para el resumen.
 - Obtener suma SHA (GNU/Linux):

```
shasum [-anumBits] Fichero.ext > Fichero.sha
```
 - Chequear suma SHA (GNU/Linux):

```
shasum -c Fichero.sha
```
 - SHA-512 es el algoritmo utilizado por defecto en GNU/Linux para guardar la contraseña.



Restricciones de tiempo (/etc/shadow)

- Introducir restricciones de tiempo o envejecimiento para la validez de la cuenta o de la contraseña.
 - **changed** ⇒ fecha del último cambio de contraseña.
 - **minlife** ⇒ nº de días que han de pasar para poder cambiar la contraseña.
 - **maxlife** ⇒ nº de días máximo que puede estar con la misma contraseña sin cambiarla.
 - **warn** ⇒ cuántos días antes de que la contraseña expire (maxlife) el usuario será informado sobre ello, indicándole que tiene que cambiarla.
 - **inactive** ⇒ nº de días después de que la contraseña expire en que la cuenta se deshabilitará si no ha sido cambiada.
 - **expired** ⇒ fecha en la que la cuenta expira y se deshabilita de forma automática.



Restricciones de tiempo

- El fichero /etc/login.defs tiene los valores por defecto.
- Comando chage (administrador):
 - `chage -d ult_día usuario` ⇒ último cambio de password.
 - `chage -m min_días usuario` ⇒ nº de días que han de pasar para poder cambiar la contraseña.
 - `chage -M max_días usuario` ⇒ nº de días máximo que puede estar con la misma contraseña sin cambiarla.
 - `chage -W warn_días usuario` ⇒ establece un aviso de que la contraseña expira un número de días antes de que expire, indicándole que tiene que cambiarla.
 - `chage -I inac_días usuario` ⇒ nº de días después de que la contraseña expire que la cuenta se deshabilitará de forma automática si la contraseña no ha sido cambiada.
 - `chage -E exp_días usuario` ⇒ fecha en la que la cuenta expira y se deshabilita de forma automática.



Restricciones de tiempo

- Supongamos que el usuario pagutierrez cambia su contraseña el 1 de marzo y **root** ejecuta estas órdenes:

```
1 chage -M 20 pagutierrez
2 chage -W 6 pagutierrez
3 chage -I 5 pagutierrez
4 chage -E 2017-10-30 pagutierrez
```

- Los tiempos quedan fijados de la siguiente manera:



Restricciones de tiempo

- Supongamos que el usuario pagutierrez cambia su contraseña el 1 de marzo y **root** ejecuta estas órdenes:

```
1 chage -M 20 pagutierrez
2 chage -W 6 pagutierrez
3 chage -I 5 pagutierrez
4 chage -E 2017-10-30 pagutierrez
```

- Los tiempos quedan fijados de la siguiente manera:
 - El 14 de marzo pagutierrez recibirá el primer aviso para que cambie su contraseña.



Restricciones de tiempo

- Supongamos que el usuario pagutierrez cambia su contraseña el 1 de marzo y **root** ejecuta estas órdenes:

```
1 chage -M 20 pagutierrez
2 chage -W 6 pagutierrez
3 chage -I 5 pagutierrez
4 chage -E 2017-10-30 pagutierrez
```

- Los tiempos quedan fijados de la siguiente manera:
 - El 14 de marzo pagutierrez recibirá el primer aviso para que cambie su contraseña.
 - El 20 de marzo, debería haber cambiado su contraseña.



Restricciones de tiempo

- Supongamos que el usuario pagutierrez cambia su contraseña el 1 de marzo y **root** ejecuta estas órdenes:

```
1 chage -M 20 pagutierrez
2 chage -W 6 pagutierrez
3 chage -I 5 pagutierrez
4 chage -E 2017-10-30 pagutierrez
```

- Los tiempos quedan fijados de la siguiente manera:
 - El 14 de marzo pagutierrez recibirá el primer aviso para que cambie su contraseña.
 - El 20 de marzo, debería haber cambiado su contraseña.
 - Si no cambia la contraseña, como se ha fijado el tiempo de inactividad, la cuenta aún no se bloqueará.



Restricciones de tiempo

- Supongamos que el usuario pagutierrez cambia su contraseña el 1 de marzo y **root** ejecuta estas órdenes:

```
1 chage -M 20 pagutierrez
2 chage -W 6 pagutierrez
3 chage -I 5 pagutierrez
4 chage -E 2017-10-30 pagutierrez
```

- Los tiempos quedan fijados de la siguiente manera:
 - El 14 de marzo pagutierrez recibirá el primer aviso para que cambie su contraseña.
 - El 20 de marzo, debería haber cambiado su contraseña.
 - Si no cambia la contraseña, como se ha fijado el tiempo de inactividad, la cuenta aún no se bloqueará.
 - Si el 25 de marzo pagutierrez no ha cambiado su contraseña, la cuenta será bloqueada.



Restricciones de tiempo

- Supongamos que el usuario pagutierrez cambia su contraseña el 1 de marzo y **root** ejecuta estas órdenes:

```
1 chage -M 20 pagutierrez
2 chage -W 6 pagutierrez
3 chage -I 5 pagutierrez
4 chage -E 2017-10-30 pagutierrez
```

- Los tiempos quedan fijados de la siguiente manera:
 - El 14 de marzo pagutierrez recibirá el primer aviso para que cambie su contraseña.
 - El 20 de marzo, debería haber cambiado su contraseña.
 - Si no cambia la contraseña, como se ha fijado el tiempo de inactividad, la cuenta aún no se bloqueará.
 - Si el 25 de marzo pagutierrez no ha cambiado su contraseña, la cuenta será bloqueada.
 - La cuenta expira, pase lo que pase, el 30 de octubre.



Ficheros de inicialización

- Directorio `/etc/skel/` \Rightarrow ficheros que se copian automáticamente a cada `$HOME`.
- Los ficheros de inicialización son *scripts* shell que realizan tareas como dar valor a variables, nombrar alias, realizar funciones específicas...
- Los ficheros dependen del intérprete de órdenes seleccionado:
 - Bourne shell: `sh`.
 - Bourne again shell: `bash`.
 - C shell: `csh`.
- Incluyen el `PATH`, variables de entorno, `umask`, funciones de inicialización, alias, var. del propio shell..



Ficheros de inicialización

Se ejecuta al hacer un <i>login</i> en el sistema por SSH o por terminal real	.bash_profile en bash .profile en bash y sh .login en csh
Cada vez que se ejecuta una shell, aunque no conlleve <i>login</i>	.bashrc en bash .cshrc en csh
Al salir del sistema el usuario (al finalizar la sesión)	.bash_logout en bash .logout en Csh



Selección de intérprete de órdenes

- En el último campo del fichero `/etc/passwd`, se establece el **intérprete de órdenes** que se ejecuta al entrar al sistema.
- En el fichero `/etc/shells` se indican los *shells* permitidos.
- Un usuario puede cambiar su *shell* con `chsh`:
 - ¡Ojo! Si se **prohíbe** un *shell*, no se podrá elegir con `chsh`, pero los usuarios que ya lo tenían asignado lo podrán seguir usando.
- Si un usuario no tiene asignado ningún intérprete de órdenes, se usará el **shell por defecto** `/bin/sh`.
- Si se desea que el usuario no pueda entrar al sistema se le puede asignar **`/bin/false`** o **`/sbin/nologin`**.
- También se puede establecer como *shell* un **fichero ejecutable**:
 - Cuando el usuario entre al sistema se ejecuta, y, al finalizar la ejecución, el usuario sale del sistema (no llega a hacer login).



Cuentas restrictivas

- Las **cuentas restrictivas** permiten limitar las acciones de los usuarios en el sistema.
- Se pueden crear de dos formas:
 - Asignar como *shell* un fichero ejecutable que realice una tarea determinada, y al terminar se sale del sistema:
 - Usuario para hacer copias de seguridad: como *shell* tiene un *script* que hace esa tarea.
 - Usuario para apagar el sistema: ejecuta la orden shutdown.
 - ★ Los usuarios restrictivos de este tipo tienen que tener los permisos necesarios para poder hacer la tarea asignada. Estos permisos se asignan a nivel de identificador de usuario.
 - ¿Qué usuario puede apagar el sistema?.



Cuentas restrictivas

- Usando el *shell* restrictivo **/bin/rbash**:
 - `rbash` es un enlace simbólico a `/bin/bash` (`rbash` es equivalente a `/bin/bash -r`).
 - Este intérprete se comporta como un intérprete normal, salvo que el usuario no puede hacer determinadas tareas, como:
 - Cambiar de directorio.
 - Establecer o modificar los valores de `$PATH` o `$HOME`.
 - Especificar nombres u órdenes que contengan `/`.
 - Usar redirección.
 - Utilizar la orden `exec` para reemplazar el *shell* por otro programa.
 - A estos usuarios **hay que limitarles los ficheros** que pueden ejecutar, copiándolos a un directorio y que su `PATH` sea sólo ese directorio. En otro caso, con un `PATH` “normal”, es casi como si no tuviesen restricciones.



Añadir un nuevo usuario al sistema

- Pasos a realizar (del 1 al 7, automatizados con herramientas):
 - ❶ Decidir el nombre de usuario, el UID, y los grupos a los que va a pertenecer (grupo **primario** y grupos **secundarios**).
 - ❷ Introducir los datos en los ficheros `/etc/passwd` y `/etc/group` (poniendo como contraseña `"*"`).
 - ❸ Asignar un *password* a la nueva cuenta.
 - ❹ Si las *shadow* están activas, escribir la contraseña.
 - ❺ Establecer los parámetros de **envejecimiento** de la cuenta.
 - ❻ Crear el directorio `$HOME` del nuevo usuario, establecer el propietario y grupo correspondiente y los permisos adecuados.
 - ❼ Copiar ficheros necesarios por defecto (`.bash_profile`, `.bashrc`...) desde `/etc/skel/`.
 - ❽ Establecer otras facilidades: *quotas*, *mail*, permisos, etc.
 - ❾ Ejecutar cualquier tarea de inicialización propia del sistema.
 - ❿ Probar la nueva cuenta.



Herramientas para crear/modificar cuentas de usuario

- Las herramientas de creación de cuentas de usuario suelen realizar todas las tareas básicas del proceso, a excepción de las específicas (quotas, impresión, etc.).
 - **adduser** o `useradd` ⇒ crear cuentas de usuario, o modificar cuentas ya existentes. Toma los valores por defecto de `/etc/default/useradd` y de `/etc/login.defs`.
`useradd` se salta algunos pasos.
 - `usermod` ⇒ modificar cuentas.
 - **deluser** o `userdel` ⇒ eliminar cuentas (por defecto no borra el directorio \$HOME).
 - `newusers` ⇒ crea cuentas de usuarios utilizando la información introducida en un fichero de texto (en *batch*), que ha de tener el formato del fichero `/etc/passwd` (no copia los ficheros de inicialización).
 - `users-admin` ⇒ herramienta en modo gráfico.



Grupos

- **Grupos:** colecciones de usuarios que comparten recursos o ficheros del sistema.
 - Características de un grupo:
 - Nombre del grupo o `groupname`.
 - Identificador del grupo (GID) \Rightarrow internamente el sistema identifica al grupo por este número.
 - **Objetivo:** Garantizar permisos concretos para un conjunto de usuarios, sin tener que aplicarlos a cada uno.
- El fichero de configuración es `/etc/group`, con el formato:
`nombre:x:gid:lista de usuarios`
 - `nombre` \Rightarrow nombre del grupo.
 - `gid` \Rightarrow identificador del grupo.
 - `lista de usuarios` que pertenecen al grupo, sep. por “,”.

P.ej., `pas:x:519:pagutierrez,jsanchezm,i22fenaf`



Grupos

- Los grupos pueden tener contraseña \Rightarrow `/etc/gshadow`:
 - Si un usuario sabe la contraseña de un grupo, puede usarlo sin pertenecer a él con la orden `newgrp`.
 - Información en `/etc/gshadow`: grupo, contraseña, usuarios administradores (pueden cambiar la contraseña y los miembros) y miembros (idea parecida al `/etc/shadow`).
- Tipos de grupos:
 - **Primarios** \Rightarrow grupo especificado en `/etc/passwd`.
 - **Secundarios** \Rightarrow otros grupos (indicados en `/etc/group`).
- Funcionamiento de los grupos:
 - Al crear un fichero se establece como grupo propietario el **grupo activo** del usuario en ese momento.
 - Grupo activo \Rightarrow grupo primario (salvo que usemos `newgrp`).
 - Al determinar los permisos sobre un fichero, se usan todos los grupos del usuario.



Grupos

- `addgroup grupo` \Rightarrow crear un nuevo grupo.
- `groupmod grupo` \Rightarrow modificar un grupo existente.
- `delgroup grupo` \Rightarrow eliminar un grupo.
- `newgrp grupo` \Rightarrow cambiar de grupo activo (lanza un shell)
- `gpasswd grupo` \Rightarrow asignar una contraseña a un grupo:
 - Si el usuario no pertenece al grupo, pero el grupo tiene contraseña, se le solicita y pasa a ser su grupo activo.
- `gpasswd -a user grupo` \Rightarrow añadir un usuario a un grupo.
- `groups [usuario]` \Rightarrow grupos a los que pertenece un usuario.
- `id [usuario]` \Rightarrow lista el identificador del usuario y los grupos a los que pertenece.
- `grpck` \Rightarrow chequea la consistencia del fichero de grupos.



Rangos del UID

- $UID \in [0, 99]$: Usuarios que representan al propio SO.
- $UID \in [100, 499]$: Usuarios especiales que representan servicios o programas.
- $UID \geq 1000$: Usuarios normales.



Algunos usuarios y grupos estándar

- **Usuarios** estándar:
 - root \Rightarrow Cuenta del administrador (0).
 - bin (utilidades comunes de usuarios, 2), daemon (ejecución de demonios, 1), lp, sync, shutdown, etc. \Rightarrow Tradicionalmente usados para poseer ficheros o ejecutar servicios
 - mail, news, ftp \Rightarrow Asociados con herramientas o facilidades.
 - postgres, mysql, xfs \Rightarrow Creados por herramientas instaladas en el sistema para administrar y ejecutar sus servicios.
 - nobody o nfsnobody \Rightarrow Usado por NFS y otras utilidades, usuario sin privilegios.



Algunos usuarios y grupos estándar

- Grupos estándar:
 - root, sys.
 - bin, daemon, adm, lp, disk, mail, ftp, nobody, etc.
 - kmem \Rightarrow Grupo propietario de los programas para leer la memoria del kernel.
 - user o users \Rightarrow Grupo de los usuarios normales (no siempre se usa).



Referencias



Evi Nemeth, Garth Snyder, Trent R. Hein y Ben Whaley
Unix and Linux system administration handbook.
Capítulo 7. *Adding new users*.
Prentice Hall. Cuarta edición. 2010.



Aeleen Frisch.
Essential system administration.
Capítulo 6. *Managing users and groups*.
O'Reilly and Associates. Tercera edición. 2002.



Programación y Administración de Sistemas

4. Gestión de usuarios

Pedro Antonio Gutiérrez

Asignatura "Programación y Administración de Sistemas"

2º Curso Grado en Ingeniería Informática

Escuela Politécnica Superior

(Universidad de Córdoba)

pagutierrez@uco.es

6 de marzo de 2017

