ADMINISTRACIÓN DE REDES Y SISTEMAS

PRÁCTICAS DE LABORATORIO

Práctica 1: Administración básica de sistemas POSIX

Esta práctica es, en buena parte, un repaso y profundización en algunos temas que ya se han aprendido en el laboratorio de Sistemas Operativos. Fundamentalmente se tratarán la gestión de usuarios, la gestión de los permisos del sistema de ficheros, el lanzamiento de servicios y la monitorización del sistema.

La práctica tiene una duración presencial de una semana, aunque se espera que el alumno trabaje adicionalmente en casa y que mejore sus conocimientos y habilidades a lo largo de las primeras semanas de la asignatura.

No es necesario entregar ninguna memoria de la práctica, aunque se recomienda que los alumnos elaboren un resumen de las actividades que van realizando, puesto que les será de utilidad para resolver las pruebas de evaluación programadas a lo largo de la asignatura.

El enunciado plantea diversas tareas que orientan a los alumnos sobre las actividades y pruebas que deberían llevar a cabo para alcanzar una buena compresión de la materia asociada a la práctica. Es responsabilidad de los alumnos el seguir esta guía como mejor les parezca, adaptándola a sus necesidades e incidiendo en los temas que les planteen mayores problemas. En todo caso, siempre contarán con el asesoramiento del profesor de laboratorio.

Gestión de cuentas de usuario (repaso obligatorio)

Toda actividad en el sistema se desarrolla asociada a alguna cuenta de usuario. En general, la cuenta de usuario puede representar a un humano (cuenta personal) o a un servicio (habitualmente implementado mediante un proceso de tipo demonio). Cada cuenta de usuario tiene un identificador numérico (UID, User Identifier) que la distingue. Para comodidad de los usuarios humanos, los UID tienen un identificador alfanumérico equivalente, al que se llama nombre de usuario.

Los usuarios adicionalmente pueden pertenecer a múltiples grupos, los cuales se identifican mediante un identificador numérico único (GID, Group Identifier) y que también tiene su correspondiente nombre de grupo.

Los UID, GID y sus correspondientes nombres son asignados discrecionalmente por el administrador del sistema, si bien ciertos rangos de identificadores comúnmente están reservados para el sistema operativo y diversas aplicaciones. Existe un identificador especial, que es el UID 0, que representa al superusuario (también llamado root). En todo caso, conviene que el administrador del sistema utilice una política clara de generación y asignación de identificadores, particularmente cuando estos identificadores han de poderse intercambiar entre un abanico de sistemas y servicios.

La información sobre los identificadores existentes, información de autenticación en base a contraseña y atributos diversos se almacena en una base de datos de usuarios y grupos. Físicamente, esta base de datos se encuentra implementada en diversos ficheros de configuración del sistema bajo el directorio /etc. Los más importantes son /etc/master.passwd, /etc/group y /etc/login.conf. Utilizando módulos diversos, también cabe la posibilidad de almacenar esta información en sistemas centralizados, como LDAP, o implementar sistemas de autenticación alternativos.

La gestión de las cuentas de usuario se puede realizar de tres maneras básicas: mediante la herramienta `pw`, mediante la herramienta `adduser` y editando manualmente diversos ficheros de configuración. El procedimiento mediante `pw` es el más recomendable de los tres y el alumno deberá conocerlo bien, pues recurrirá a él con frecuencia a lo largo de la asignatura y se considera **esencial** para superar las pruebas de evaluación de la asignatura.

Tarea 1: Cree una cuenta de usuario utilizando la herramienta `pw useradd`, especificando la opción `-m` para que también se cree un directorio personal para la cuenta y póngale una contraseña con la opción `-h0` o bien mediante el mandato `passwd`. Observe las líneas que se han añadido a la base de datos de usuarios y grupos.

Después de crear las primeras cuentas se debe comprobar su buen funcionamiento. Para ello puede abrirse una sesión SSH con la cuenta en cuestión. Al menos deberían realizarse las siguientes comprobaciones.

1. Abrir una sesión de trabajo con el nombre y contraseña establecidos.
2. Verificar con el mandato `id` que la identidad del usuario es la esperada (nombre, UID, grupos, GID).
3. Verificar con el mandato `pwd` que el directorio de entrada al sistema es el esperado.
4. Verificar con el mandato `ls -ld $HOME` que los permisos del directorio base del usuario son los adecuados.
5. Verificar con el mandato `ls -lA` que el contenido del directorio base del usuario es el esperado y con los permisos adecuados.
6. Verificar que el usuario puede crear ficheros en su directorio personal.

Tarea 2: Acceda con la cuenta nueva mediante SSH y compruebe su buen funcionamiento.

Tarea 3: Cree varios usuarios de prueba adicionales. En alguno de los casos puede definir algún valor concreto para un atributo, como el UID, el intérprete de mandatos o la descripción.

Tarea 4: Bloquee una de las cuentas y compruebe que no puede abrir sesión con ella.

Tarea 5: Modifique la contraseña de una de las cuentas.

Tarea 6: Cambie la descripción de una cuenta.

Tarea 7: (Opcional) Cambie el UID de una cuenta. Compruebe que este cambio no es trivial.

Tarea 8: Elimine alguna de las cuentas.

Gestión de grupos de usuario (repaso obligatorio)

Tarea 9: Cree un grupo llamado “amigos”.

Tarea 10: Cree varios usuarios más, haciendo que pertenezcan al grupo “amigos”.

Tarea 11: Meta en el grupo amigos a alguna cuenta ya existente.

Tarea 12: Saque del grupo amigos a alguno de sus miembros.

Supervisión de usuarios (opcional)

En este apartado se observará la salida de diversos programas que proporcionan información acerca de los usuarios del sistema.

Tarea 13: Pida a algunos compañeros de laboratorio que abran una sesión de trabajo en su máquina y abra usted mismo varias sesiones de trabajo en diferentes consolas virtuales de su máquina. A continuación utilice los mandatos `who`, `w`, `finger` y `last` y observe la información que proporciona cada uno de ellos.

Variables de entorno (opcional)

Muchas aplicaciones utilizan variables de entorno para obtener parámetros de funcionamiento[1]. Estas variables se suelen definir en los ficheros de arranque del intérprete de mandatos. Ahí también se pueden definir otros aspectos del perfil del usuario, como alias, opciones para el intérprete de mandatos, programas que han de ejecutarse automáticamente al abrir una sesión de trabajo, etc.

Para los intérpretes de la familia `sh`, estos ficheros son `/etc/profile` y `~/.profile`. Para los intérpretes de la familia `csh`, los ficheros en cuestión son `/etc/csh.cshrc`, `/etc/csh.login`, `~/.cshrc` y `~/.login`; además, algunos intérpretes (por ejemplo, `bash`) pueden usar ficheros adicionales (ver su página de manual para más detalles).

Los ficheros del directorio `/etc` son comunes para todos los usuarios y son el lugar adecuado para almacenar las definiciones que se deseen aplicar de manera universal, es decir, a todos los usuarios que abran una sesión de trabajo. Los ficheros ubicados en el directorio del usuario sólo le son aplicables a él y normalmente se ejecutan después de los del sistema, por lo que pueden sustituir o complementar algunas de las definiciones universales contenidas en los ficheros del sistema. Esta organización de ficheros no es exclusiva de los intérpretes de mandatos, sino que otras muchas aplicaciones suelen tener un fichero de configuración maestro que leen primero y después ficheros propios de cada usuario que leen después y que permiten complementar al fichero común para todos los usuarios.

En los intérpretes de la familia `sh`, las variables de entorno se definen mediante la sentencia "`VARIABLE=valor ; export VARIABLE`". En la familia `csh` se definen mediante "`setenv VARIABLE valor`", excepto algunas variables, como `path` y `prompt`, que se definen mediante `set variable=valor`. Para más detalles, consulte la documentación de estos intérpretes de mandatos o pregunte al profesor de laboratorio.

Importante: tal y como se distribuye el sistema operativo FreeBSD, las cuentas que se crean con `pw` y `adduser` copian desde el directorio `/usr/share/skel` diversos ficheros de arranque en el directorio personal de cada usuario. Esos ficheros definen una serie de variables de entorno habituales que machacan a las definiciones universales de los ficheros de arranque del directorio `/etc`.

Tarea 14: ¿Qué es lo que habría que modificar en su sistema para poner las siguientes definiciones de variables de entorno para un cierto usuario?

Variable	Valor	Propósito
EDITOR	ee	Indica el editor preferido que deberían usar todas las aplicaciones que necesiten que el usuario edite algún fichero. `ee` es un editor de texto fácil de usar.
PAGER	less	Indica el programa preferido que deberían usar todas las aplicaciones que deseen visualizar su salida página a página. `less` es un buen visualizador.
LESS	-I	Indica las opciones por defecto del programa `less`. La opción `-I` indica que no diferencie entre mayúsculas y minúsculas al hacer búsquedas.
LANG	es_ES.UTF-8	Indica el idioma, configuración nacional y juego de caracteres preferido.
CLICOLOR	Ninguno. Definirla	Indica al mandato `ls` que utilice colores para diferenciar

[1] Las páginas de manual de los programas suelen tener al final un apartado llamado ENVIRONMENT en el que se listan todas las variables de entorno que el programa entiende.

Variable	Valor	Propósito
	sin más.	los tipos de ficheros.
PATH	Los que haya y además /usr/local/prog/bin.	Indica la lista de directorios en los que se buscarán los ejecutables. En `sh` se define como `PATH=d1:d2:d3`, y en `csh` se define como `set path=(d1 d2 d3)`. Pueden usarse las expresiones `$PATH` o `$path` (respectivamente) para incluir la lista definida en estos momentos.
PS1	\h:\w \$	Define la cadena que mostrará el intérprete de mandatos para pedir nuevas órdenes. Este ejemplo muestra el nombre de la máquina, el directorio actual y el carácter $ para los usuarios normales o el carácter # para el superusuario. Nota: este ejemplo es sólo para el intérprete de mandatos `bash`. No olvide encerrar la cadena entre apóstrofes.
prompt	%m:%/%#	Análoga a la variable PS1, pero para el intérprete de mandatos `tcsh`. No olvide encerrar la cadena entre apóstrofes.

Nota: para ver todas las variables de entorno definidas puede ejecutar "`export -p`" en `sh` y "`setenv`" en `csh`. Si lo desea, puede comprobar el efecto de la variable `EDITOR` ejecutando los mandatos `vipw`, `chsh` o cualquier otro que lance un editor externo. Para comprobar el efecto de la variable `PAGER` puede utilizar el mandato `man` y buscar una página de manual cualquiera. Para comprobar el efecto de la variable `PATH` escriba y compile en el directorio `/usr/local/prog/bin` un programa que muestre la frase "hola mundo". Para comprobar el efecto de la variable de entorno `LANG` use `ls -l` y compruebe cómo aparecen las fechas, por ejemplo.

Tarea 15: ¿Qué es lo que habría que hacer en su sistema para que las definiciones de variables de entorno anteriores surtieran efecto para todos los usuarios del sistema, tanto si usan `sh` como si usan `csh`?

La decisión de si las definiciones universales de variables de entorno es mejor ponerlas en los ficheros de `/etc` de los intérpretes de mandatos o bien en los ficheros adecuados de `/usr/share/skel` puede verse influida por diversos factores, pero como norma general se recomienda ponerlas en los ficheros de `/etc`, de forma que dichas definiciones estén en un único sitio y los usuarios no tengan que preocuparse de ellas. Ello lleva apareada la ventaja de que si hay que cambiarlas, basta con tocar un solo fichero de arranque y no todos los ficheros de arranque de los usuarios. Además, estarán más protegidas de posibles borrados accidentales por parte de los usuarios. Finalmente, si un usuario realmente desea tener sus propias definiciones de variables de entorno, nada le impide meterlas en sus ficheros de arranque particulares.

Es importante destacar que, tal como viene el sistema operativo de serie, el contenido del directorio `/usr/share/skel` es un obstáculo para aprovechar las ventajas de los ficheros de configuración globales de `/etc`, lo cual se espera que el alumno haya podido apreciar.

Permisos de ficheros (repaso obligatorio)

El sistema de ficheros alberga toda la información que se maneja en el sistema. Es imprescindible que exista un mecanismo de control de acceso eficaz y para ello es fundamental conocer bien las propiedades de los permisos de ficheros estándar de POSIX.

Todo fichero (en sentido amplio, incluyendo a ficheros normales, directorios, enlaces, etc) tiene asociados un dueño, grupo y permisos estándar. El dueño de un objeto es inicialmente el usuario que lo creó, aunque el superusuario puede cambiar posteriormente el dueño del objeto. Únicamente el dueño del objeto y el superusuario pueden definir los derechos de acceso a un objeto.

Para decidir si se autoriza un acceso a un objeto del sistema de ficheros, el núcleo compara las credenciales del proceso que solicita el acceso con los derechos de acceso definidos para el objeto. Las credenciales constan de varios UID (real, efectivo, guardado) y GID (real, efectivo, guardado). Como caso especial, si el acceso es solicitado por un proceso con UID efectivo de superusuario, el sistema siempre lo autoriza.

Con frecuencia se desea que ciertos usuarios o procesos sean capaces de intercambiar información y trabajar cooperativamente por medio de directorios y ficheros compartidos. Eso normalmente se consigue por medio de la utilización de grupos de usuario y definiendo adecuadamente la máscara de permisos por defecto (`umask`) para que los nuevos ficheros y directorios compartidos se creen con unos permisos adecuados.

Tarea 16: Con un usuario cualquiera, compruebe el valor actual de `umask` y cree un fichero y un directorio. Observe los permisos con que se han creado.

Tarea 17: Cambie el valor de `umask` y cree más ficheros y directorios. Observe los permisos con que se han creado.

Tarea 18: Compruebe cómo se puede cambiar el valor de `umask` de manera permanente para un usuario (opcional, se realiza en los mismos ficheros del apartado de las variables de entorno).

Tarea 19: Cree el directorio /home/pruebas, de manera que cualquier usuario pueda acceder a él. El resto de ejercicios se harán dentro de este directorio.

Tarea 20: Cree los usuarios adan, bea, carla y david. Cree los grupos chicos (para adan y david) y chicas (para bea y carla).

Tarea 21: Cree los directorios alfa, beta, gamma y delta. El directorio alfa permitirá acceso total al usuario adan, y ningún otro usuario o grupo podrá acceder. El directorio beta permitirá acceso total a la usuaria bea y de solo lectura al grupo chicas. El directorio gamma permitirá acceso total a la usuaria carla y al grupo chicos. El directorio delta permitirá acceso total al usuario david y de solo lectura a cualquier otro usuario y grupo.

Tarea 22: Abriendo sesiones con los usuarios anteriores, compruebe que se satisfacen los controles de acceso deseados.

Tarea 23: Cree ficheros y directorios nuevos en cada uno de los directorios y compruebe si tras crearlos tienen los controles de acceso deseados. En caso negativo, decida cuál sería el valor de `umask` más adecuado en cada caso.

Tarea 24: En el directorio gamma, cree un fichero f1.txt que únicamente tenga permisos para carla. A continuación, intente borrarlo con el usuario david. ¿Qué sucede?

Tarea 25: En el directorio gamma, cree un fichero f2.txt con permiso de lectura para todo el mundo. Compruebe que la usuaria bea no puede ver el contenido de ese fichero. ¿Qué es lo mínimo que hay que hacer para que la usuaria bea sea capaz de acceder al fichero f2.txt?

Ejecución de operaciones privilegiadas (repaso obligatorio)

En todos los sistemas operativos existen ciertas operaciones privilegiadas (por ejemplo, apagar la máquina, subir la prioridad de los procesos, saltarse los permisos del sistema de ficheros, etc.)

que sólo puede ejecutar el administrador o bien determinados usuarios a los que el administrador ha otorgado los privilegios necesarios.

Los sistemas Unix convencionales aprovechan los grupos de usuario y los bits setuid y setgid para controlar la ejecución de determinados programas privilegiados. Otra posibilidad es utilizar el mandato `su` o la herramienta `sudo`.

Tarea 26: Observe el dueño, grupo y permisos del programa `/sbin/shutdown`. ¿Quién puede ejecutarlo? ¿Cómo puede hacer que un usuario normal tenga permiso para parar el sistema?

Opcional: puede buscar otros programas con los bits setuid y setgid activados ejecutando el mandato "`find / -perm +6000 -ls`".

El mandato `su` permite convertirse en otro usuario y se utiliza más frecuentemente para convertirse en superusuario. En los sistemas BSD sólo pueden convertirse en superusuario los usuarios pertenecientes al grupo `wheel`.

Tarea 27: Utilice el mandato `su` para que un usuario cambie su identidad.

En muchos casos no es aconsejable que un usuario normal pueda convertirse en superusuario mediante `su`, ya que eso le permitirá realizar cualquier operación en la máquina. Por ello, existe una herramienta llamada `sudo` que permite controlar qué mandatos privilegiados podrá ejecutar un usuario normal.

Tarea 28: Compruebe si está instalado el programa `sudo` (`pkg info`). En caso necesario, instálelo (`pkg install sudo`). A continuación, haga que el usuario `pepe` pueda ejecutar cualquier mandato mediante `sudo`.

Si se utiliza `sudo` para permitir mandatos privilegiados a usuarios que no sean de mucha confianza, se ha de ser muy consciente de las consecuencias para la seguridad del sistema que esto puede tener.

Tarea 29: (Opcional) Suponga que, utilizando `sudo`, el administrador permite que el usuario juan edite con `vi` un fichero protegido. ¿Cómo podrá juan aprovechar este permiso para hacerse superusuario? ¿De qué manera segura puede usarse sudo para editar ficheros que requieren privilegios?

Programación de tareas periódicas (opcional)

En un sistema Unix se puede programar que una tarea se ejecute con cierta periodicidad (mandato `crontab`) o bien que se ejecute una única vez en una fecha y hora concretas (mandato `at`).

Tarea 30: Programe una tarea periódica que se ejecute cada 2 minutos y haga algo fácilmente observable; por ejemplo, matar todos los procesos de un determinado usuario, borrar todos los ficheros del directorio `/var/tmp` cuyo nombre comience por "a", añadir una línea a un fichero de texto… Recuerde que cualquier mensaje que pueda generar la tarea por stdout/stderr no aparecerá en la pantalla, sino en el buzón del usuario al que pertenece.

Mensajes de syslog (obligatorio)

En ocasiones una aplicación necesita comunicar que ha sucedido algo, desde una indicación meramente informativa hasta un evento catastrófico para la aplicación. Cuando la aplicación no es interactiva, como es el caso de los demonios y demás procesos que proporcionan servicios del sistema, el método habitual es generar un mensaje a tal efecto por medio del mecanismo `syslog`. Configurando adecuadamente el demonio `syslogd`, el administrador del sistema puede seleccionar qué tipos de mensajes son de su interés y qué se debe hacer con ellos. De cara

a las pruebas de evaluación de la asignatura, se considera **esencial** que el alumno sea capaz de consultar y manejar el sistema de `syslog`.

Para hacer pruebas en las siguientes cuestiones, es interesante saber que el programa `logger(1)` permite generar manualmente mensajes para `syslog`. Por ejemplo, para generar un mensaje del subsistema "mail", con importancia "notice" y con el nombre de programa "correo" se usaría la orden `logger -t correo -p mail.notice "Hola mundo"`.

Tarea 31: Examine la configuración actual del demonio `syslogd` (fichero `/etc/syslog.conf`) y ojee el contenido de algunos de los ficheros de `/var/log`, como `messages`, `cron` o `security`.

Tarea 32: Suponga que una cierta aplicación llamada "acme" genera mensajes para el subsistema "local4". Se desea registrar todos los mensajes de importancia "warning" o superior generados por dicha aplicación en el fichero `/var/log/acme.log` ¿Qué líneas habría que añadir al fichero de configuración de `syslogd`?

Rotación de ficheros históricos (opcional)

El programa `newsyslog(8)` se ejecuta periódicamente y su misión es rotar los ficheros históricos, con la posibilidad de comprimir los ficheros antiguos y eliminarlos cuando alcancen una determinada antigüedad. Este programa se configura mediante el fichero `/etc/newsyslog.conf`. Aunque se suele usar para rotar ficheros generados desde `syslogd`, este programa es válido para rotar cualquier otro tipo de fichero.

Tarea 33: Observe el fichero `/etc/crontab` y compruebe con qué periodicidad se ejecuta el programa `newsyslog`.

Tarea 34: ¿Qué línea habría que añadir al fichero `/etc/newsyslog.conf` para que el fichero `/var/log/acme.log` se rote una vez por semana, comprimiéndose las versiones anteriores y guardándose los datos de las últimas doce semanas?