Administración de sistemas POSIX

Gestión de Usuarios

En el sistema se puede decir que existen dos tipos de procesos de usuario:

- **Procesos vinculados a un terminal**: se lanzan desde una sesión de terminal, en respuesta a las órdenes de un usuario humano.
- **Demonios**: se lanzan al arrancar el sistema y responden a las peticiones recibidas desde otros procesos, generalmente a través de la red.

Todo proceso que se ejecuta en el sistema pertenece a algún usuario y podemos hablar de tres categorías fundamentales:

- Procesos vinculados a una sesión de terminal: se ejecutan bajo la identidad del usuario que abrió la sesión, el cual suele tener una cuenta de **usuario normal** y corriente.
- Demonios privilegiados: ofrecen algún servicio para el que se necesita poder acceder a cualquier recurso de la máquina (por ejemplo, un servidor de ficheros). Se ejecutan bajo la identidad del **superusuario**.
- Demonios no privilegiados: ofrecen un servicio para el que solo se necesita acceder a ciertos recursos (por ejemplo, un servidor de DHCP). Se ejecutan bajo la identidad de un **pseudousuario** (una cuenta de usuario normal, pero que existe solo para este servicio y no para un usuario humano).

Por motivos de seguridad, la mayoría de los servicios se ejecutan bajo la identidad de un pseudousuario. Aquellos que se ejecutan bajo la identidad del superusuario tratan de pasar a un usuario normal siempre que sea posible.

Las cuentas de usuario en FreeBSD se administran mediante diversas herramientas, entre las que destaca el programa **pw**. Estas herramientas manipulan diversos ficheros de /etc y en algunos casos también el directorio particular de cada cuenta.

En ocasiones algunos usuarios han de poder realizar operaciones privilegiadas. Para ello hay tres mecanismos fundamentales en la mayoría de los sistemas de tipo POSIX:

- Hacer que el usuario pertenezca a un grupo que tenga permisos sobre los directorios o ficheros necesarios.
- Ejecutar la acción por medio de un programa **setuid/setgid**.
- Ejecutar la acción por medio del programa **sudo**

Practica 1

Pw

1. **Creación de usuario "pepe"**:

    - Se utiliza el comando **pw useradd pepe -m -h0** para crear el usuario "pepe" con su directorio personal y sin asignar una contraseña de forma explícita.

      - pw useradd pepe -m -h0

    - Se utiliza **pw usershow pepe** para mostrar la información del usuario "pepe", que incluye su UID, GID, directorio personal y shell predeterminado.

      - pw usershow pepe

    - Se utiliza **pw groupshow pepe** para mostrar la información del grupo de "pepe".

      -pw groupshow pepe

2. **Inicio de sesión y comprobaciones para el usuario "pepe"**:

    - Se inicia sesión mediante SSH con el usuario "pepe".
    - Se utiliza **id** para verificar la identidad del usuario.
    - Se utiliza **pwd** para mostrar el directorio de trabajo actual.
    - Se verifica los permisos del directorio personal de "pepe" con **ls -ld $HOME**.
    - Se crea un archivo de prueba y se verifica con **ls -l**.

      -ssh pepe@localhost

      -id

      -pwd

      --ls -ld $HOME

      -touch prueba1.txt

      -ls -l

3. **Creación de usuarios "luis" y "ana"**:

    - Se utiliza **pw useradd -D -M 700** para establecer el valor predeterminado de los permisos de los directorios creados.

      -pw useradd -D -M 700

    - Se utiliza **pw useradd luis -m -h0 -c 'Luis Carvajal' -s /bin/csh** para crear el usuario "luis" con un shell específico.

      -pw useradd luis -m -h0 -c 'Luis Carvajal' -s /bin/csh

    - Se utiliza **pw useradd ana -m -h0 -c 'Ana García'** para crear el usuario "ana" con una descripción.

-pw useradd ana -m -h0 -c 'Ana García'

4. **Creación del usuario "beatriz" y cambio del shell predeterminado**:
    - Se crea el usuario "beatriz" con un UID específico mediante **pw useradd beatriz -m -h0 -u 20001**.

      -pw useradd beatriz -m -h0 -u 20001

      -ssh beatriz@localhost
    - Se cambia el shell predeterminado a **/usr/local/bin/bash** utilizando **pw useradd -D -s /usr/local/bin/bash**.
    - Se crea el usuario "carlos" con la herramienta **pw useradd carlos -m -h0**.

      -pw useradd -D -s /usr/local/bin/bash

      -pw useradd carlos -m -h0
5. **Bloqueo de la cuenta "ana" y cambio de contraseña para "beatriz"**:
    - Se bloquea la cuenta de "ana" con **pw lock ana**.

      -pw lock ana
    - Se intenta iniciar sesión con "ana" para verificar que está bloqueada.

      -ssh ana@localhost
    - Se desbloquea la cuenta de "ana" con **pw unlock ana**.

      -pw unlock ana
    - Se cambia la contraseña de "beatriz" utilizando **passwd beatriz**.

      -passwd beatriz
6. **Modificación de la descripción para el usuario "ana"**:
    - Se utiliza **pw usermod ana -c 'Ana Jiménez'** para cambiar la descripción del usuario "ana".

      -pw usermod ana -c 'Ana Jiménez'
    - Se utiliza -**finger ana** para verificar el cambio en la descripción.
7. **Cambio del UID para el usuario "ana" y comprobación de los permisos del directorio personal**:
    - Se utiliza **pw usermod ana -u 5000** para cambiar el UID de "ana".

      -pw usermod ana -u 5000

      -ssh ana@localhost

      -id

-ls -ld $HOME

8. **Eliminación de los usuarios "carlos" y "beatriz"**:
    - Se elimina el usuario "carlos" utilizando **pw userdel carlos**.
    - Se elimina el usuario "beatriz" utilizando **rmuser -y beatriz**.
9. **Creación del grupo "amigos" y asignación de usuario "emilio"**:
    - Se crea el grupo "amigos" con **pw groupadd amigos**.
    - Se añade el usuario "emilio" al grupo "amigos" con **pw useradd emilio -m -h0 -G amigos**.
    - Se inicia sesión con "emilio" para verificar la asignación al grupo.

      -ssh emilio@localhost
10. **Añadir usuario "pepe" al grupo "amigos" y verificar los cambios**:
    - Se añade "pepe" al grupo "amigos" con **pw usermod pepe -G amigos**.
    - Se muestra la información del grupo "amigos" con **pw groupshow amigos**.
11. **Añadir usuario "luis" al grupo "amigos" y verificar los cambios**:
    - Se añade "luis" al grupo "amigos" con **pw groupmod amigos -m luis**.
    - Se muestra la información actualizada del grupo "amigos" con **pw groupshow amigos**.
12. **Eliminar usuario "pepe" del grupo "amigos" y verificar los cambios**:
    - Se elimina "pepe" del grupo "amigos" con **pw groupmod amigos -d pepe**.
    - Se muestra la información actualizada del grupo "amigos" con **pw groupshow amigos**.

Supervisión de Usuarios

13.

- who # Muestra quién está conectado al sistema y desde dónde
- w # Proporciona una visión más detallada de los usuarios conectados, incluyendo qué están haciendo
- finger # Muestra información detallada sobre los usuarios conectados, incluyendo su nombre, terminal, hora de inicio de sesión, entre otros detalles
- last # Muestra un historial de las últimas conexiones al sistema, incluyendo la hora de inicio y fin de sesión, así como la duración de la sesión

Estos comandos proporcionarán información valiosa sobre la actividad de los usuarios en el sistema.

Para las Tareas 14 y 15, que involucran la definición de variables de entorno para un usuario específico y para todos los usuarios del sistema, respectivamente, se necesitan modificar ciertos archivos de configuración. Aquí están los pasos:

Tarea 14:

1. Para definir las variables de entorno para un usuario específico (por ejemplo, "usuario"):

Edite el archivo .profile o .bash_profile del usuario en su directorio de inicio

nano ~/.profile # O ~/.bash_profile

2. Agregue las siguientes líneas al archivo con los valores deseados:

bashCopy code

export EDITOR=ee

export PAGER=less

export LESS=-I

export LANG=es_ES.UTF-8

export CLICOLOR

export PATH=$PATH:/usr/local/prog/bin

export PS1='\h:\w \$'

Tarea 15:

1. Para definir las variables de entorno para todos los usuarios del sistema:

Edite el archivo de perfil global del shell correspondiente

nano /etc/profile # Para shell sh/bash

nano /etc/csh.cshrc # Para shell csh/tcsh

Permisos de Ficheros

- **r** (read): Permite la lectura del archivo o directorio. (4)
- **w** (write): Permite la escritura en el archivo o directorio, o bien la creación o eliminación de archivos dentro de un directorio. (2)
- **x** (execute): Permite la ejecución de un archivo (en el caso de un archivo regular) o la búsqueda y acceso a archivos dentro de un directorio (en el caso de un directorio). (1)
- **-** (guion): Indica que el permiso correspondiente está desactivado.

    - **rwxr-xr- -**

1. **rwx**: Permisos del propietario del archivo. (Primer Dígito)
2. **r-x**: Permisos del grupo al que pertenece el archivo. (Segundo Dígito)
3. **r--**: Permisos para otros usuarios del sistema. (Tercer Dígito)

   Cada dígito se descompone en tres bits, que pueden tener un valor de 0 a 7, representando diferentes combinaciones de permisos:

Ejercicio Repaso Permisos

Sea un sistema FreeBSD en el que están definidos, entre otros, los siguientes usuarios:

- pperez, que es miembro de los grupos “usuarios” y “pperez”.
- jgarcia, que es miembro de los grupos “wheel" y “usuarios”.
- asanchez, que es miembro de los grupos “operator” y “proyecto”.

Estos son los permisos de algunos ficheros y directorios del sistema de ficheros:

Fichero	**Dueño**	**Grupo**	**Permisos**
/	root	wheel	775
/home	root	usuarios	775
/home/pperez	pperez	usuarios	711
/home/pperez/p1	pperez	usuarios	707
/home/jgc	jgarcia	proyecto	644
/home/jgc/bin	bin	bin	755
/home/jgc/bin/cc1	root	bin	600
/datos	pperez	wheel	772
/datos/script2.sh	asanchez	usuarios	555
/datos/f2.dat	asanchez	proyecto	777

1. **ppperez: cd /home/pperez**
    - Verdadero. Se necesita permiso de ejecución (x) en todos los directorios desde la raíz hasta el directorio al que se quiere cambiar, y pperez lo tiene en /, /home y /home/pperez.
2. **pperez: p1 (Ejecutar el programa p1)**
    - Verdadero. Se necesita permiso de lectura y ejecución en /home/pperez/p1, y pperez lo tiene.
3. **pperez: cp /home/pperez/p1 /home/jgc**
    - Falso. El usuario pperez puede leer el fichero de origen (p1), pero no puede pasar ni escribir en /home/jgc.
4. **jgarcia: cd /home/pperez**
    - Verdadero. El usuario jgarcia necesita permiso de paso (x) en todos los directorios desde la raíz hasta /home/pperez, y lo tiene, aunque no puede listar los ficheros en /home/pperez debido a la falta de permiso de lectura.
5. **jgarcia: cat /home/pperez/p1**
    - Falso. El usuario jgarcia pertenece al grupo usuarios, pero no tiene permiso de lectura en /home/pperez/p1.
6. **jgarcia: rm /home/pperez/p1**
    - Falso. El usuario jgarcia no tiene permiso de escritura en /home/pperez, por lo que no puede borrar el fichero.
7. **jgarcia: chmod 777 /home/pperez**
    - Falso. Solo el dueño de un fichero/directorio (o el superusuario) puede cambiar sus permisos.
8. **asanchez: cd /home/jgc/bin**
    - Falso. El usuario asanchez necesita permiso de paso en /home/jgc/bin, pero no lo tiene.
9. **asanchez: cp /home/jgc/bin/cc1 /datos**
    - Falso. El usuario asanchez necesita permiso de paso en /datos, pero no lo tiene.
10. **pperez: cp /home/pperez/p1 /datos/script3.sh**
    - Verdadero. El usuario pperez tiene permiso de lectura en /home/pperez/p1 y de escritura en /datos.
11. **asanchez: cp /home/pperez/p1 /datos/script3.sh**
    - Falso. El usuario asanchez necesita permiso de paso en /datos, pero no lo tiene.

Tarea 16: Comprobar el valor actual de umask y crear un archivo y un directorio.

umask # Comprobar el valor actual de umask

touch archivo.txt # Crear un archivo

mkdir directorio # Crear un directorio

ls -l archivo.txt directorio # Ver los permisos de los archivos creados

Tarea 17: Cambiar el valor de umask y crear más archivos y directorios.

umask 022 # Cambiar el valor de umask

touch archivo2.txt # Crear otro archivo

mkdir directorio2 # Crear otro directorio

ls -l archivo2.txt directorio2 # Ver los permisos de los archivos creados

Tarea 18: Cambiar el valor de umask de manera permanente para un usuario.

Editar el archivo de perfil del usuario correspondiente nano ~/.profile # o ~/.bash_profile # Agregar la línea siguiente (o modificar la existente): umask 022 # Cambiar umask a 022 # Guardar y cerrar el archivo, luego cerrar y volver a abrir la sesión para que los cambios surtan efecto.

Tarea 19: Crear el directorio /home/pruebas y establecer permisos adecuados.

sudo mkdir /home/pruebas # Crear el directorio

sudo chmod 777 /home/pruebas # Establecer permisos para que cualquier usuario pueda acceder

Tarea 20: Crear usuarios y grupos.

sudo pw useradd adan

sudo pw useradd bea

sudo pw useradd carla

sudo pw useradd david

sudo pw groupadd chicos

sudo pw groupadd chicas

sudo pw groupmod chicos -m adan,david

sudo pw groupmod chicas -m bea,carla

Tarea 21: Crear los directorios con los permisos especificados.

cd /home/pruebas

mkdir alfa beta gamma delta

chmod 700 alfa # Permisos solo para el usuario adan

chmod 770 beta # Permisos para bea y el grupo chicas

chmod 770 gamma # Permisos para carla y el grupo chicos

chmod 744 delta # Permisos para david, solo lectura para otros

sudo chown adan:chicos /home/pruebas/alfa

sudo chmod 700 /home/pruebas/alfa

sudo chown bea:chicas /home/pruebas/beta

sudo chmod 770 /home/pruebas/beta

sudo chown carla:chicos /home/pruebas/gamma

sudo chmod 770 /home/pruebas/gamma

sudo chown david /home/pruebas/delta

sudo chmod 744 /home/pruebas/delta

Tareas 22 a 25: Estas tareas requieren interacción manual y observación de los resultados. Puedes utilizar los comandos **su** o **ssh** para iniciar sesión con los usuarios correspondientes y realizar las comprobaciones necesarias. Por ejemplo:

su adan ls -l /home/pruebas/alfa touch /home/pruebas/alfa/archivo_prueba.txt ls -l /home/pruebas/alfa exit # Salir de la sesión de adan

Ejecución de operaciones privilegiadas

Tarea 26:

ls -l /sbin/shutdown

Este comando muestra los permisos del programa **/sbin/shutdown**, lo que nos permite determinar quién puede ejecutarlo y bajo qué condiciones.

-sr-xr-- 2 root operator 15616 Apr 7 2023 /sbin/shutdown

El propósito de esta tarea es comprender cómo se gestionan los privilegios de ejecución de programas en sistemas Unix y cómo se puede otorgar acceso a un usuario normal para ejecutar operaciones privilegiadas, como apagar el sistema.

1. **Observar los permisos de /sbin/shutdown:** Ejecutamos el comando **ls -l /sbin/shutdown** para ver los permisos del programa **shutdown**. Si el bit setuid (**s**) está establecido en el propietario del archivo (**root**), entonces el programa se ejecutará con los privilegios del propietario, en este caso, del usuario **root**. Por lo tanto, solo el usuario **root** (o cualquier usuario que pueda actuar como **root**) puede ejecutar el programa **shutdown**.
2. **Conceder acceso a un usuario normal para apagar el sistema:** Para permitir que un usuario normal ejecute el comando **shutdown**, podemos usar **sudo**. **sudo** permite a los usuarios ejecutar comandos con los privilegios de otro usuario (generalmente **root**). Para ello, debemos agregar al usuario normal al archivo **/etc/sudoers** utilizando el comando **visudo**. Por ejemplo, para permitir que el usuario "pepe" ejecute el comando **shutdown**, podemos agregar la siguiente línea al archivo **/etc/sudoers**:

pepe ALL=(ALL) /sbin/shutdown

Esto permitirá al usuario "pepe" ejecutar **shutdown** con **sudo**, lo que le permitirá apagar el sistema.

Tarea 27:

El propósito de esta tarea es entender cómo utilizar el comando **su** para cambiar la identidad de un usuario.

1. **Utilizar el comando su para cambiar de identidad:** El comando **su** permite a un usuario cambiar su identidad a la de otro usuario. Por ejemplo, si el usuario actual es "pepe" y queremos cambiar a "otro_usuario", podemos ejecutar:

   su otro_usuario -> Nos pedirá contraseña de ese usuario

Tarea 28:

pkg info sudo # Verifica si sudo está instalado

pkg install sudo # Instala sudo si es necesario

El primer comando verifica si **sudo** está instalado en el sistema, mientras que el segundo comando instala **sudo** si no está presente. Luego, para permitir que el usuario **pepe** ejecute cualquier comando con **sudo**, se edita el archivo de configuración de **sudo** utilizando el comando **visudo** y se agrega la línea **pepe ALL=(ALL) ALL**

Syslog

1. **Facilidades:** Las facilidades son categorías predefinidas que indican la fuente del mensaje. Permiten clasificar los mensajes según su origen. Algunos ejemplos comunes de facilidades son:
    - **auth**: Para mensajes relacionados con autenticación y autorización.
    - **mail**: Para mensajes relacionados con el sistema de correo electrónico.
    - **local0** a **local7**: Reservados para aplicaciones específicas que deseen registrar mensajes.

Estas facilidades proporcionan una manera estructurada de identificar la fuente de los mensajes registrados, lo que facilita la filtración y el análisis posterior.

2. **Prioridades:** Las prioridades son niveles de importancia asignados a los mensajes. Permiten clasificar los mensajes según su gravedad o urgencia. Van desde el nivel más bajo, "debug" (depuración), hasta el nivel más alto, "emergency" (emergencia). La lista completa de prioridades estándar incluye:
    - **debug**: Mensajes de depuración, útiles para desarrolladores.
    - **info**: Mensajes informativos que no indican problemas.
    - **notice**: Mensajes normales pero significativos.
    - **warning**: Mensajes que indican condiciones potencialmente problemáticas.
    - **err**: Mensajes que indican errores no críticos.
    - **crit**: Mensajes que indican errores críticos.
    - **alert**: Mensajes que requieren atención inmediata.
    - **emergency**: Mensajes que indican condiciones de emergencia que requieren acción inmediata.

Al asignar una prioridad a un mensaje, se comunica la importancia relativa del evento registrado, lo que ayuda a los administradores del sistema a priorizar y responder adecuadamente a los problemas.

3. **Acciones:** Las acciones determinan qué se hace con los mensajes registrados. Pueden incluir:
    - **Impresión en la consola**: Los mensajes pueden mostrarse en la consola del sistema para que los usuarios y administradores puedan verlos directamente.
    - **Almacenamiento en archivos de registro específicos**: Los mensajes se pueden escribir en archivos de registro específicos, como **/var/log/messages**, **/var/log/auth.log**, etc.

- **Envío por correo electrónico**: Los mensajes pueden enviarse por correo electrónico a direcciones de correo electrónico predefinidas.
- **Reenvío a otro servidor syslog**: Los mensajes pueden reenviarse a otro servidor syslog para su procesamiento y almacenamiento centralizados.

Estas acciones permiten a los administradores del sistema configurar syslog para que los mensajes se manejen de la manera más adecuada según las necesidades del entorno y los requisitos de seguimiento y resolución de problemas.

Tarea 31: Para examinar la configuración actual del demonio **syslogd** y ojear el contenido de algunos de los ficheros de **/var/log**, puedes ejecutar los siguientes comandos:

cat /etc/syslog.conf # Examinar la configuración actual de syslogd

cat /var/log/messages # Ojear el contenido del archivo de mensajes generales del sistema

cat /var/log/cron # Ojear el contenido del archivo de registros de cron

cat /var/log/security # Ojear el contenido del archivo de registros de seguridad

Estos comandos te permitirán ver la configuración actual del demonio **syslogd** en el archivo **/etc/syslog.conf** y ojear el contenido de algunos de los archivos de registro en **/var/log**, como **messages**, **cron** o **security**, que son comunes en muchos sistemas Unix y Linux.

Tarea 32: Para registrar todos los mensajes de importancia "warning" o superior generados por la aplicación "acme" en el fichero **/var/log/acme.log**, se pueden añadir las siguientes líneas al fichero de configuración de **syslogd**:

- local4.warning /var/log/acme.log

Esto indicará al demonio **syslogd** que registre todos los mensajes del subsistema **local4** con importancia **warning** o superior en el archivo **/var/log/acme.log**

Copias de Seguridad

Tipos de copias de seguridad

Por los ficheros que se incluyen:

- **Totales:** Copia todos los archivos y directorios seleccionados en su totalidad.
- **Parciales:** Solo copia los archivos y directorios que han cambiado desde la última copia de seguridad.
- **Incrementales:** Copia solo los archivos que han cambiado desde la última copia de seguridad incremental, es decir, desde la última copia completa o incremental.
- **Diferenciales:** Similar a las copias incrementales, pero en lugar de copiar solo los archivos que han cambiado desde la última copia, copia los archivos que han cambiado desde la última copia completa.

Sistemas alternativos:

- **Instantáneas:** Captura el estado de un sistema de archivos en un momento dado, permitiendo revertir el sistema al estado capturado en caso de pérdida de datos.
- **Clonación:** Crea una copia idéntica de un disco o sistema completo, lo que permite restaurar el sistema en su totalidad en caso de fallo.

Medios:

Los medios utilizados para almacenar las copias de seguridad pueden incluir cintas, CD/DVD, discos duros y dispositivos de almacenamiento extraíbles como unidades USB.

Por el destino:

- **Locales:** Las copias de seguridad se almacenan en dispositivos de almacenamiento conectados directamente al sistema.
- **Remotas:** Las copias de seguridad se almacenan en sistemas remotos, a menudo a través de redes.

Herramientas para copias de seguridad

Herramientas comunes:

- **dump/restore:** Utilidades tradicionales de UNIX para realizar copias de seguridad y restauración de sistemas de archivos.
- **tar, cpio, compresores (gzip, bzip2, xz):** Utilidades para crear archivos de copia de seguridad y comprimirlos.
- **rsync:** Utilidad para sincronizar y hacer copias de seguridad de archivos y directorios de manera eficiente.
- **Bacula, Amanda:** Son sistemas de gestión de copias de seguridad que ofrecen funcionalidades avanzadas para programar, administrar y recuperar copias de seguridad en entornos complejos.

Ejemplos de uso de herramientas

Ejemplos de uso de **tar**:

- **Hacer una copia: tar czf copia.tgz /home** crea una copia comprimida del directorio **/home**.
- **Restaurar una copia: tar xzpf copia.tgz** restaura la copia previamente creada en el directorio actual.

Ejemplos de uso de **dump/restore**:

- **Hacer una copia: dump 0af copia.dump /home** crea una copia de seguridad del sistema de archivos **/home**.
- **Restaurar una copia: restore -ivf /dev/nsa0** restaura la copia previamente creada desde el dispositivo **/dev/nsa0**.
-

Arranque y parada de servicios

Manejo de servicios en FreeBSD

- **Configuración:** Se realiza en el archivo **/etc/rc.conf**, donde se pueden habilitar o deshabilitar servicios utilizando la variable **nombreservicio_enable="YES/NO"**. Se pueden especificar flags adicionales usando **nombreservicio_flags="-x -y -z"**. Para configurar estas opciones, se puede usar el comando **sysrc nombreservicio_enable=YES**.
- **Arranque y parada:** Los scripts de inicio y parada de servicios se encuentran en los directorios **/etc/rc.d** y **/usr/local/etc/rc.d**. Para manejar los servicios, se utiliza el comando **service SERVICIO start/stop/restart/status/...**.

Programación de tareas

Crontab

- Se utiliza para programar tareas que se ejecutarán periódicamente.
- La sintaxis es **minutos horas día_mes mes día_semana mandato**.
- Por ejemplo, ***/5 * * * * wget -q -O /dev/null http://moodle/cron.php** ejecuta el comando **wget** cada 5 minutos para acceder a una URL y ejecutar un script.
- **30 23 * * 6 find /var/tmp –mtime +7 -delete** elimina archivos del directorio **/var/tmp** que tengan más de 7 días de antigüedad todos los sábados a las 11:30 PM.

At, atq, atrm, atrun

- Herramientas para programar tareas para ejecutarse una vez en el futuro. **at** programa una tarea para ejecutarse en un momento específico, **atq** muestra las tareas en cola,

atrm elimina tareas de la cola y **atrun** es el demonio que ejecuta las tareas en el momento especificado.

Monitorización del sistema

Objetivos de la monitorización

- Detectar y solucionar problemas a tiempo.
- Averiguar las causas de los problemas.
- Afinar el comportamiento del sistema.
- Predecir tendencias de uso futuras.
- Vigilar la seguridad del sistema.
- Justificar la toma de decisiones.

Tipos de monitorización

- **Datos históricos:** Analizan el comportamiento del sistema en el pasado.
- **Datos en tiempo real:** Monitorizan el sistema en tiempo real para detectar problemas inmediatos.
- **Auditoría de seguridad:** Vigilan la seguridad del sistema y detectan actividades sospechosas.
- **Pruebas periódicas:** Realizan comprobaciones regulares del sistema para garantizar su correcto funcionamiento.
- **Inspección física:** Monitorizan aspectos físicos del sistema, como impresoras, cableado, estado de los equipos, etc.

Monitorización y ajuste del sistema

- **Datos históricos:** Se recopilan y analizan mediante herramientas como **syslogd**, **vmstat**, **iostat**, **fstat**, **pstat**, **netstat**, **nfsstat**, **ipcs**, entre otras.
- **Datos en tiempo real:** Se monitorizan utilizando las mismas herramientas que en datos históricos, pero se observan en tiempo real.
- **Paquetes de software:** Se utilizan herramientas de monitorización como Nagios, Zabbix y Snort.
- **Ajuste:** Se ajusta el sistema según las necesidades utilizando **sysctl** y recompilación del núcleo.

Este es un resumen detallado de los conceptos y herramientas relacionadas con el arranque y parada de servicios, la programación de tareas y la monitorización del sistema en un entorno FreeBSD.