SERVICIOS Y PROCESOS

1.PROCESOS LINUX

Los procesos se pueden definir como programas que están corriendo en nuestro Sistema Operativo. Frecuentemente se les refiere como "tareas". El contexto de un programa en ejecución es lo que se llama un proceso. Este contexto puede ser más procesos hijos que se hayan generado del principal (proceso padre), recursos del sistema que este consumiendo, atributos de seguridad, etc.

- Linux=SO multitarea y multiusuario (Múltiples procesos pueden operar simultáneamente sin interferirse unos con los otros.)

- Programas y procesos son entidades distintas: En un SO multitarea, múltiples instancias de un programa pueden ejecutarse simultáneamente. (Si cinco usuarios desde equipos diferente ejecutan el mismo programa al mismo tiempo, habría cinco procesos distintos del mismo programa.)

- Organización jerárquica: Procesos padre y procesos hijo, Init es el primer proceso que se ejecuta tras el Inicio. Todos los procesos en GNU/Linux son hijos de init.

¿Cómo se pueden clasificar los procesos?

◦ Procesos Normales:

Generalmente son lanzados en una terminal (tty) y corren a nombre de un usuario. Son los programas que utiliza el usuario generalmente y se encuentran conectados a una terminal. El programa aparecerá en la pantalla y interactuará con el usuario.

◦ Procesos Daemon

Corren a nombre de un usuario y no tienen salida directa por una terminal, es decir, corren en 2º plano. Generalmente los conocemos como servicios. En vez de usar la terminal para escuchar un requerimiento, lo hacen a través de un puerto.

◦ Procesos Zombie

Son aquellos que han completado su ejecución pero aún tiene una entrada en la tabla de procesos. El proceso hijo no recibió una señal por parte del proceso de nivel superior (proceso padre) que lo creó, informándole que su vida útil ha terminado. Se pueden deber a errores de programación, situaciones no contempladas por el programador y generalmente provocan lentitud y/o inestabilidad en el Sistema.

¿En qué estados se pueden encontrar los procesos?

◦ Running ® En ejecución
◦ Sleeping (S) Esperando su turno para ejecutarse
◦ Stopped (D) Esperan a que se finalice alguna operación de Entrada/Salida
◦ Zombie (Z) Han terminado, pero siguen apareciendo en la tabla de procesos

¿Cómo podemos ver el árbol de procesos?

- **Directorio /proc** Estructura que generada por nuestro kernel durante el arranque. Nos muestra lo que está ocurriendo con nuestro kernel.

Comandos para ver estado de los procesos

- **Comando top** Monitoriza dinámicamente los procesos del sistema mostrando su estado, uso de CPU, cantidad de memoria, tiempo desde su inicio, nombre, etc.

2.ADMINISTRACIÓN DE PROCESOS

- **Comando ps** Muestra los procesos que están ocurriendo en el sistema. No es interactivo. Saca una foto de los procesos que están corriendo en ese mismo momento. Tiene una gran cantidad de opciones, las cuales varían dependiendo del estilo en el que se use el comando:

  Estilo UNIX, donde las opciones van precedidas por un guión –
  Estilo BSD, donde las opciones no llevan guión
  Estilo GNU, donde se utilizan nombres de opciones largas y van precedidas por doble guión - -

En el listado de procesos aparecen varias columnas, las principales son:

- **p o PID:** Process ID, número único o de identificación del proceso.
- **P o PPID:** Parent Process ID, padre del proceso
- **U o UID:** User ID, usuario propietario del proceso
- **t o TT o TTY:** Terminal asociada al proceso, si no hay terminal aparece un '?'
- **T o TIME:** Tiempo de uso de cpu acumulado por el proceso
- **c o CMD:** El nombre del programa o camndo que inició el proceso
- **RSS:** Resident Sise, tamaño de la parte residente en memoria en kb
- **SZ o SIZE:** Tamaño virtual de la imagen del proceso
- **NI:** Nice, valor nice (prioridad) del proceso, un número positivo significa menos tiempo de procesador y negativo más tiempo(-19 a 19)
- **C o PCPU:** Porcentaje de cpu utilizado por el proceso
- **STIME:** Starting Time, hora de inicio del proceso
- **S o STAT:** Status del proceso, estos pueden ser los siguientes:
  -R runnable, en ejecución, corriendo o ejecutándose
  -S sleeping, proceso en ejecución pero sin actividad por el momento, o esperando por algún evento para continuar
  -T sTopped, proceso detenido totalmente, pero puede ser reiniciado
  -Z zombie, difunto, proceso que por alguna razón no terminó de manera correcta, no debe haber procesos zombies
  -D uninterruptible sleep, son procesos generalmente asociados a acciones de IO del sistema
  -X dead, muerto, proceso terminado pero que sigue apareciendo, igual que los Z no deberían verse nunca

Las opciones completas de ps se encuentran en la terminal:

- man ps (manual)
- ps L
- ps –e (muestra todos los procesos)
- ps –ef (muestra opciones completas)
- ps –eF (muestra opciones completas extras)
- ps ax (formato BSD sin guion, a muestra todos, x sin mostrar tty)
- ps aux (formato BSD sin guion, u muestra usuarios y demás columnas)
- ps –eo user,pid,tty (-o output personalizado, campos separados por coma)
- pstree (procesos en forma de árbol; pstree –help te da las opciones más comunes)
- ps –eH (muestra árbol de procesos)
- ps axf (lo mismo en formato BSD)
- ps –ec (el comando se está ejecutando, sin la ruta, solo el nombre real)
- ps –el (muestra formato largo varias columnas)
- ps L (no muestra procesos, lista todos los códigos de formatos)

- **Comando top** Permite el monitoreo en tiempo real del estado de los procesos y otras variantes del sistema**.** Es interactivo y por defecto se actualiza cada 3 seg.

- **Comando kill** Sirve para enviar señales (signals) a los procesos. Sintáxis: kill PID (siendo PID el número ID del proceso)
  - kill-l (lista todas las posibles señales que pueden enviarse a un proceso)

  SIGHUP 1 Para y reinicia el proceso. Utilizado para servicios
  SIGKILL 9 Finalización inmediata del proceso, sin esperar su finalización natural
  SIGCONT 18 Reactiva un proceso tras su suspensión con SIGSTOP
  SIGSTOP 19 Suspende un proceso y lo deja inactivo (pausa)

Podemos mandar las señales invocándolas a través del número de la señal o de su código, por ejemplo: (kill -9 11428) mata proceso

- **Comando killall** Funciona de manera similar a kill, pero con al diferencia de en vez de indicar un PID se indica el nombre del programa. Afectará a todos los procesos que tengan ese nombre.

- **Comando nice** Permite cambiar la prioridad de un proceso cuando se inicia su ejecución. Prioridades van de -20 (la más alta) a 19 (la más baja). Con la opción ps –l es posible observar la columna NI que muestra ese valor. Sin argumento, devuelve la prioridad por defecto. -n -5 inicia con prioridad de -5

- **Comando renice** Permite cambiar la prioridad de un proceso en tiempo real, sin necesidad de detener el proceso (el campo NI en el primer caso en -5, y en el segundo con renice quedó en 7, en tiempo real)

- **Comando &** Se utiliza cuando ejecutamos procesos en segundo plano (background). Permite liberar la terminal de un programa que se espera que dure un tiempo considerable ejecutándose, mandándolo a segundo plano o background, con objeto de liberar la terminal y continuar trabajando.

- **Comandos fg (foreground) y bg (background)** Permiten manipular procesos que estén suspendidos temporalmente

- **Comando Jobs** Nos permite listar los procesos actuales en ejecución

CONFIGURACIÓN DE USUARIOS Y PERMISOS

1.USUARIOS Y GRUPOS

Linux=SO multiusuario. Varios usuarios pueden conectarse y trabajar en él de forma simultánea

¿Qué caracteriza a un usuario?

- Login necesario para conectarse al sistema
- Password o contraseña
- Conjunto de datos adicionales (domicilio, teléfono, etc.)

root

Es el usuario con más privilegios en Linux. Único con derechos para crear o eliminar a otros usuarios. Derecho para acceder a todo el sistema de ficheros sin ninguna restricción.

Grupos de usuarios

Permiten otorgar los mismos privilegios a un conjunto de usuarios. Siempre que se añada un usuario al sistema, se creará un grupo con su mismo nombre, llamado grupo primario. Se podrá incorporar el usuario a otros grupos secundarios.

Identificación de usuarios y grupos de usuarios

Ambos se identifican a través de un identificador númerico (ID). Ambos se identifican a través de un identificador numerico (ID). Usuario root siempre es el ID cero. Cada usuario posee:

- Identificador de usuario asociado (uid)
- Uno o varios identificadores de grupo (gid)

FICHEROS ESPECIALES

/etc/passwd

Recoge la información asociada a los usuarios. Consta de 7 campos separados por el símbolo dos puntos (:)

- **Nombre:** Indica el nombre de la cuenta de usuario
- **Contraseña:** Indica una X siempre, sería un fallo de seguridad muy grande almacenarlo aquí.
- **UID:** Numero que se la asigna a cada usuario en el sistema
- **GID**: Número del grupo principal al que pertenece este usuario. (El grupo principal suele tener el mismo nombre del usuario)
- **Información:** Varios campos separados por coma (título informativo, sin importancia)
- **Dir. Personal:** Directorio donde se almacenará el perfil del usuario. Habitualmente es /home/Nombredelusuario
- **Shell:** Programa que se debe ejecutar para ofrecerle un terminal de acceso al sistema a este usuario.

/etc/shadow

Recoge las contraseñas y datos afines. Este fichero consta de 8 campos separados por el símbolo dos puntos (:)

- **Nombre:** Indica el nombre de la cuenta de usuario
- **Contraseña:** Contraseña del usuario encriptada. Representada en 3 partes distintas separadas con el símbolo del dólar ($).
    - Id. Esto nos indica el método de encriptación que fue utilizado. Un valor de 1 indica MD5, un valor de 2 Blowfish, 3 es NT Hash, 5 es SHA-256 y 6 es SHA-512.
    - Salt. Este valor se usa por los algoritmos de encriptación y puede ser de hasta 16 caracteres. Este valor se crea aleatoriamente en cada generación de contraseña.
    - Hash. La contraseña en sí misma. MD5 usa 22 caracteres, SHA-256 usa 43, y SHA-512 usa 86.

Si nos encontramos con un asterisco en este campo significa que la cuenta está bloqueada. Lo mismo se puede conseguir colocando una exclamación (!) al principio de la contraseña. Una cuenta sin contraseña la veremos o bien con todo este campo en blanco, o bien con dos exclamaciones consecutivas (!!).

- **Ult. Cambio:** Indica el día en que se cambió la contraseña por última vez. (Este número indica los días que han transcurrido desde el 1-1- 1970).
- **Mínimo:** El mínimo número de días que hay que esperar para cambiar la contraseña
- **Máximo**: El máximo número de días antes de que la contraseña caduque.
- **Aviso:** Antes de que la contraseña caduque, avisaremos al usuario al llegar a este número de días.
- **Inactivo:** Una vez que la contraseña caduque esperaremos este número de días antes de bloquearla definitivamente.
- **Caducidad:** Es una fecha, en la cual la cuenta automáticamente quedará bloqueada. (Este número indica los días que han transcurrido desde el 1-1-1970).

/etc/group

Recogen las información de los grupos y sus miembros Este fichero consta de 4 campos separados por el símbolo dos puntos (:)

- **Nombre:** Indica el nombre del grupo
- **Contraseña:** Indica una X siempre, sería un fallo de seguridad muy grande almacenarlo aquí. Se almacenaría en el fichero /etc/gshadow
- **GID:** Es el identificador numérico del grupo
- **Miembros:** Los nombres de los miembros del grupo separados por coma.

/etc/gshadow

Recogen las información de los grupos, sus miembros y contraseñas. Este fichero consta de 4 campos separados por el símbolo dos puntos (:)

- **Nombre:** Indica el nombre del grupo
- **Contraseña:** Una contraseña encriptada con las mismas características que la existente en el fichero shadow. Si existe una contraseña en un grupo, un usuario que se quiera introducir como miembro del grupo tendrá que saber dicha contraseña ya que el sistema se la pedirá (comando newgrp). Si un grupo no tiene una contraseña, solo el root o los administradores del grupo podrán asignar miembros a dicho grupo.
- **Administradores:** Una lista de usuarios separados por coma que pueden añadir o eliminar miembros al grupo.
- **Miembros:** Una lista de usuarios separados por coma. Estos son los usuarios que pueden acceder al grupo sin que se le pida la contraseña. Esta lista debe ser la misma que la que se encuentra en /etc/group.

Otros ficheros importantes:

/etc/default/useradd: contiene los valores por defecto a la hora de añadir un usuario al sistema con el comando useradd.
/etc/adduser.conf: contiene los valores por defecto a la hora de añadir un usuario al sistema con el comando adduser.
/etc/deluser.conf: contiene los valores por defecto a la hora de eliminar un usuario del sistema con el comando deluser.
/etc/shells: el fichero de shells contiene una lista de shells válidas. Si la Shell de usuario no está en este fichero no puede acceder al sistema mediante login.
/etc/skel: directorio que almacena el contenido del directorio de los nuevos usuarios que se vayan añadiendo al sistema.

ADMINISTRACIÓN DE USUARIOS Y GRUPOS

1.GESTIÓN DE USUARIOS

CREAR USUARIOS useradd. El fichero /etc/passwd contiene los usuarios del sistema. Permite añadir un usuario indicando como parámetros la información particular para crear el usuario en la misma línea de comandos. La sintaxis es:

useradd [opciones] nombre-usuario

Entre las opciones más destacables tenemos:

-c comentarios introduce un comentario para tener más información del usuario.
-d home_directory directorio de trabajo del usuario. Suele ser /home/nombre-usuario.
-s shell intérprete de comandos (shell) del usuario. Suele ser /bin/bash.
-g gid grupo base del usuario (el grupo debe existir previamente).

-m crea el directorio home para el nuevo usuario.
-G otros_grupos lista de grupos secundarios de los que el usuario es miembro. Los grupos se separan por coma, sin espacio en blanco. Los grupos deben existir. Si el usuario es miembro de un grupo que no esté en esta lista, se borrará del grupo.
-u uid sirve para especificar un UID al usuario. Debe ser un número positivo y único. Por defecto, el sistema asigna el primer UID libre.
Ejemplo: sudo useradd -g profesores -d /home/pedro -m -s /bin/bash pedro

De esta manera habremos creado al usuario pedro y su carpeta home. Si no utilizamos la opción -m, no se creará la carpeta home del usuario; en tal caso tendríamos que crearla manualmente.
Tan solo nos quedará establecer su contraseña y para ello usaremos el siguiente comando.

passwd. Permite modificar la clave (password) de un usuario dado. La sintaxis es: passwd [usuario]

MODIFICAR USUARIOS usermod. El fichero /etc/passwd contiene los usuarios del sistema. Permite cambiar el nombre del usuario, su carpeta home, su intérprete de comandos, los grupos a los que pertenece y algunos otros parámetros. La sintaxis es:

usermod [opciones] nombre-usuario

Entre las opciones más destacables tenemos:
-c comentarios modifica el comentario del usuario.
-d home_directory modifica el directorio de trabajo del usuario.
-s shell modifica el intérprete de comandos (shell) del usuario.
-g grupo_base modifica el grupo base del usuario (debe existir previamente).
-G otros_grupos lista de grupos secundarios de los que el usuario es miembro. Los grupos se separan por coma, sin espacio en blanco. Los grupos deben existir. Si el usuario es miembro de un grupo que no esté en esta lista, se borrará del grupo.
-u uid cambio de identificador de usuario. Este valor debe ser único. No admite valores negativos. Cualquier uid de fichero que esté en el directorio origen del usuario se cambiará automáticamente al nuevo uid, pero no pasa así con los ficheros que estén en otro directorio, que deberán ser cambiados manualmente (usando chmod, chgrp, chown).
-l nombre_login cambio de login.
-L bloquea la contraseña, deshabilitando la cuenta de usuario.
-U desbloquea la contraseña, habilitando la cuenta de usuario para iniciar sesión.
Ejemplo: sudo usermod –d /home/carpeta_pedro –g grupo_base_pedro pedro

ELIMINACION USUARIOS userdel. El fichero /etc/passwd contiene los usuarios del sistema. Elimina la cuenta de usuario indicada. La sintaxis es:

userdel [opciones] nombre-usuario

Entre las opciones más destacables tenemos: **-r** elimina el directorio personal del usuario especificado.

OTROS id Muestra el UID, GID del grupo por defecto y los GID de los otros grupos a que pertenece el usuario indicado. Si no se especifica el usuario se utiliza el usuario actual. La sintaxis es:

id [nombre-usuario]

2.GESTION DE GRUPOS

CREACION DE GRUPOS groupadd. El fichero /etc/group contiene los grupos del sistema. Permite añadir un grupo de usuarios vacío indicando como parámetro el nombre del grupo. La sintaxis es:

groupadd [opciones] nombre-grupo

Entre las opciones más destacables tenemos:

-g gid especifica el GID del grupo que se va a crear.
-o indica que el grupo puede tener asociado más de un GID.
-r crea un grupo de sistema.
Ejemplo: sudo groupadd –g 500 alumnos

MODIFICACIÓN DE GRUPOS groupmod. El fichero /etc/group contiene los grupos del sistema. Permite modificar el nombre de un grupo o el GID del mismo. La sintaxis es:

groupmod [opciones] nombre-grupo

Entre las opciones más destacables tenemos:
-n cambia el nombre del grupo.
-g cambia el GID.
Ejemplo: sudo groupmod –n docentes –g 2000 profesores

ELIMINACION DE GRUPOS groupdel. El fichero /etc/group contiene los grupos del sistema. Elimina un grupo de usuarios creado previamente. La sintaxis es:

groupdel nombre-grupo

Ejemplo: sudo groupdel alumnos

CAMBIAR GRUPOS DE UN USUARIO newgrp. El fichero /etc/group contiene los grupos del sistema. Cuando se accede al sistema siempre se hace con el mismo grupo. Pero un usuario puede pertenecer a más de un grupo aunque sólo está activo uno. El usuario puede necesitar cambiarlo por otro de los que pertenezca para crear nuevos ficheros. Para ello usará el comando que nos ocupa. Ha de ser el usuario desde su login quien ejecute esta operación. Su sintaxis es:

newgrp nuevo-grupo

AÑADIR USUARIOS A UN GRUPO adduser. Permite añadir un usuario del sistema a un grupo que debe existir previamente. La sintaxis es:

adduser nombre-usuario nombre-grupo

Ejemplo: sudo adduser andres profesores

Nota: este comando también sirve para dar de alta (crear) usuarios en el sistema. Esta utilidad se verá más adelante (parecido a useradd).

QUITAR USUARIOS deluser. Permite eliminar un usuario de un grupo que debe existir previamente. La sintaxis es:

deluser nombre-usuario nombre-grupo

Ejemplo: sudo deluser andres profesores

Nota: este comando también sirve para eliminar usuarios del sistema (parecido a userdel). Esta utilidad se verá más adelante.

OTROS COMANDOS DE GRUPOS groups. Lista los grupos a los que pertenece el usuario. La sintaxis es:

groups nombre-usuario

Ejemplo: groups andres

3.GESTION DE PERMISOS SOBRE DIRECTORIOS Y FICHEROS

MODIFICION DE PERMISOS chmod. Permite modificar los permisos de uno o varios ficheros. Su sintaxis es:

chmod mascara fichero/s

Notación octal:

- El bit de ejecución añade 1 a la suma.
- El bit de escritura añade 2 a la suma.
- El bit de lectura añade 4 a la suma.

Si quisiéramos expresar lo mismo mediante una máscara simbólica, pondríamos: chmod u=rw, g=rx, o=w practica
La máscara simbólica está formada por tres códigos:

a) Clases de usuarios:

u propietario
g grupo
o otros
a todos (propietario, grupo y otros)

b) Permisos:

Símbolo Significado
r lectura
w escritura
x ejecución

c) Operación:

Símbolo Significado
+ añadir permisos
- quitar permisos
= asignar permisos

- **CAMBIO DEL PROPIETARIO DE UN FICHERO**

chown. Este comando cambia el propietario de un fichero. La sintaxis es: sudo chown nuevo_propietario fichero/s
Ejemplo: chown pepe carta

- **CAMBIO DE GRUPO A UN FICHERO**

chgrp. Este comando cambia de grupo a un fichero. La sintaxis es: sudo chgrp nuevo_grupo fichero/s

Ejemplo: chgrp usuarios1 carta1 carta2

COMANDOS PARA REALIZAR TAREAS BÁSICAS DE CONFIGURACIÓN DE SISTEMA

TAREAS BASICAS DE CONFIGURACION DEL SISTEMA

¿Qué son las tareas programadas de Windows?

Son la configuración para que un archivo, programa o proceso, se ejecute en un tiempo determinado y bajo condiciones especificadas por los usuarios.

¿Cómo se realiza el proceso en Linux?

A través de la terminal y mediante unos comandos esenciales: cron y crontab

CRON Y CRONTAB EN LINUX

¿Qué es Cron?

Su nombre proviene de la expresión griega "chronos" y su significado es tiempo. Es uno de los demonios o "daemon" (proceso en segundo plano) más importantes y habituales en el sistema. Su ejecución comienza desde el primer instante de arranque.

¿Cuál es la función principal de Cron?

Encargarse de lanzar las tareas programadas en fechas específicas y de forma automática y repetitiva. La definición de tareas se localiza en el archivo /etc/crontab. Su funcionamiento es simple, verifica si existen tareas (jobs) para su ejecución de acuerdo al horario del sistema. Se puede inicializar usando los directorios /etc/init.d o etc/rc.d/ y en cada minuto realiza un chequeo de los /etc/crontab o var/spool/cron ubicando posibles ejecuciones pendientes.

¿Qué es Crontab?

Es tan simple como un archivo de texto. Su contenido especifica una lista de todos los scripts a ser ejecutados por el sistema, así como las fechas, horas y los permisos de ejecución de los mismos. Cada usuario tiene su propio archivo crontab, y el que está ubicado en el directorio /etc, es propiedad del usuario root. Crontab es la forma más simple para administrar las tareas de cron en sistemas de tipo multiusuario. Cada asterisco representa una fracción de tiempo que determina el momento de la ejecución, seguido del usuario bajo el cual se realizará la ejecución y por último el comando a ejecutar.

Administración de los Jobs del Cron (comandos básicos)

Si queremos modificar el archivo actual usamos: crontab –e
Para obtener la lista de todas las tareas en crontab: crontab –l
Para eliminar el actual crontab del sistema: crontab –r

Formato de las tareas (jobs):

Si se deja un asterisco, quiere decir "cada" minuto, hora, día de mes, mes o día de la semana

* * * * * /bin/ejecutar/script.sh

- Cada minuto
- De cada hora
- De cada día del mes
- De cada mes
- De cada día de la semana

Palabras reservadas à Facilitan el uso de cron y Crontab

@reboot: se ejecuta una única vez al inicio.
@yearly/@annually: ejecutar cada año.
@monthly: ejecutar una vez al mes.
@weekly: una vez a la semana.
@daily/@midnight: una vez al día.
@hourly: cada hora.

CONFIGURACIÓN DE PERFILES LOCALES DE USUARIO

PERFILES DE USUARIO LINUX

Los SO están preparados para que cada usuario pueda disponer de su propia configuración y forma de trabajo dentro del sistema al que accede. Por lo general, hay características comunes y otras que pueden ser configuradas por el usuario y por el superusuario del sistema. Para cualquier usuario dado de alta en el sistema se creará una carpeta dentro del directorio /home con el nombre del usuario que contendrá el perfil que se le aplicará cuando inicie sesión en Linux. El usuario será el único que tendrá todos los derechos de uso.

Por seguridad, el usuario root tiene su propio perfil local de usuario ubicado en el directorio /root. Se pueden crear scripts y ficheros de inicio que se ejecutarán al entrar en sesión un usuario, de manera que podamos configurar el perfil de trabajo del usuario dentro del sistema.

El fichero /etc/profile

Contiene el perfil igual para todos los usuarios. En su interior podemos poner comandos que se ejecutarán al iniciar sesión cualquier usuario. También ejecuta todos los script que se encuentran en el directorio /etc/profile.d Cada vez que se inicia sesión de un usuario con el comando, se ejecutarán los siguientes ficheros ocultos (llevan el identificativo del punto) relacionados con el perfil de acceso al sistema de un usuario:

/etc/profile Ejecutar el perfil genérico para todos los usuarios
/home/nombre_usuario/.profile Ejecuta el .bashrc
/home/nombre_usuario/.bashrc Contiene comandos que se ejecutan al inicio del Shell de forma interactiva
/home/nombre_usuario/.bash_history Almacena el histórico de comandos que introduce el usuario en consola
/home/nombre_usuario/.bash_logout Se ejecuta cuando el usuario sale de la sesión

Hay que destacar que cuando se inicia sesión desde un terminal para cambiar de usuario solamente se ejecuta el fichero .bashrc

Algunas operaciones con ficheros de perfil:

carlos@sistemalinux:~$ ls -lrta | grep .profile Busca los ficheros .profile en la ubicación actual
carlos@sistemalinux:~$ ls -a Lista todos los ficheros ocultos
carlos@sistemalinux:~$ more .profile Visualizamos el contenido del fichero .profile
carlos@sistemalinux:~$ nano .profile Editamos el fichero .profile
carlos@sistemalinux:~$ su – nombre_usuario Cambiamos de usuario y se ejecuta su .profile
carlos@sistemalinux:~$ su nombre_usuario Cambiamos de usuario pero no ejecuta el .profile

Debemos tener en cuenta que cada vez que modifiquemos el fichero .profile y ejecutemos el comando su-Nombre_usuario, se ejecutará lo que éste contenga, ya que es un script de inicio de sesión del usuario.

VARIABLES DE ENTORNO

VARIABLES DE ENTORNO

¿Qué son las variables de entorno?

Son valores parametrizados que contienen información acerca del sistema y del usuario que inició sesión actualmente. Dichos valores afectan a los programas o procesos que se ejecutan en un servidor. Su uso:

Proporciona un entorno de ejecución a los programas, de forma que éstos puedan adaptarse a distintas plataformas y usuarios. Flexibiliza los programas y los ficheros por lotes, gracias a la inclusión de valores parametrizados en lugar de valores concretos.

La imagen de cada proceso incluye una copia de las variables de entorno definidas. Cuando se lanza un programa desde un intérprete de comandos, el shell copiará las variables de entorno de su imagen a la imagen de proceso creado para la ejecución del programa. El nombre de las variables de entorno en GNU/Linux es "case sensitive"

CLASIFICACION

1. Variables de entorno persistentes a nivel de SISTEMA (system-wide): Se definen en /etc/environment y serán visibles por cualquier proceso.

2. Variables de entorno persistentes a nivel de SESIÓN (sesión-wide): Se definen en ~/.pam_environment o ~/.bashrc y Serán visibles por cualquier proceso propiedad del usuario.

3. Variables de entorno (no persistentes) a nivel de intérprete de comandos: Serán visibles desde el propio shell y desde cualquier proceso lanzado por éste. El comando export de bash permite crear variables de entorno a nivel del intérprete de comandos. Las variables definidas en bash sin export se denominan variables locales.

GESTIÓN DE VARIABLES EN BASH

Comando export: export nombre de variable="valor"
Crea variables de entorno a nivel del bash actual, asignándoles de forma opcional un valor inicial. El shell bash define, de forma automática, una serie de variables locales y variables de entorno (a nivel del shell), a las que denomina Variables del shell: HOME, USER, PWD, PATH,...

Comando printenv:
Sin parámetros: Lista el nombre y valor de todas las variables de entorno (variables persistentes a nivel de sesión y a nivel de sistema, así como variables no persistentes a nivel del intérprete de comandos).
Con parámetros: printenv <nombre de variable de entorno> Envía el valor de <nombre de variable de entorno> a la salida estándar.

Comando echo: Permite mostrar el valor de una variable de entorno. echo $<nombre de variable>

VARIABLES DEL SHELL BASH

Variable de entorno PATH, y ejecución de programas desde la línea de comandos: El shell intentará ejecutar cualquier cadena que el usuario introduzca a través del prompt. Para ello realizará una serie de comprobaciones.

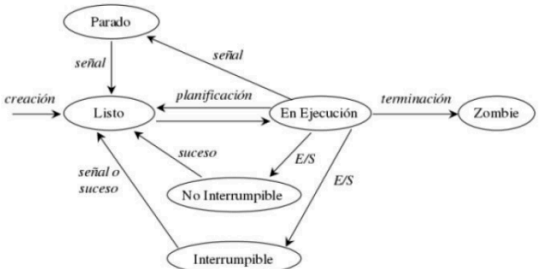
- Si se ha especificado la ruta absoluta o relativa del fichero a ejecutar, lanzará directamente su ejecución.
- Si no se ha especificado la ruta absoluta o relativa del fichero a ejecutar, realizará las siguientes comprobaciones:
  - Comprobará si se trata de un alias. En caso afirmativo, realizará la sustitución correspondiente (solo si no se ha efectuado ya anteriormente), pasando a evaluarse de nuevo por el Shell.
  - Comprobará si se trata de un comando interno. En caso afirmativo, lo ejecutará.
  - En caso de que no se trate de un comando interno, comprobará si se trata de un comando externo o de cualquier otro programa almacenado en disco. Para ello: Buscará en los directorios especificados en la variable de entorno PATH.

MONITORIZACIÓN DE SUCESOS

Una de las áreas más importantes de la administración de sistemas es la monitorización y control de los procesos. La principal razón de esto es la necesidad de mantener nuestras aplicaciones y recursos en óptimas condiciones. Muchos de estos procesos y sucesos que se dan en el sistema se ejecutan de forma "oculta", en segundo plano, y hace que la detección de sus inconsistencias pueda convertirse en una tarea algo compleja.
Por ello, dentro de los sistemas GNU/Linux existen una gran variedad de herramientas que nos van a permitir realizar una monitorización adecuada y a su vez, realizar auditorías que puedan detectar amenazas que han pasado desapercibidas para nuestros ojos.

MONITORIZACIÓN

Al contrario que con los ficheros, los procesos en GNU/Linux no pueden cambiar de propietario o grupo, ya que su vida se limita al tiempo que dura su ejecución. Aunque durante la ejecución de éstos, su estado va cambiando en función de las circunstancias que se den. Los estados por los que pueden pasar los procesos de nuestro sistema GNU/Linux son los siguientes:



- D - Ininterrumpible
- R - En ejecución
- S - Interrumpible, esperando a un evento
- T - Parado
- X - Acabado, no suele verse pero puede darse mientras se refresca el sistema
- Z - Zombie, está acabado pero aún está vinculado a su proceso padre

Cada uno de estos procesos, es identificado por una serie de datos que son generados a partir de su creación. Estos datos asociados son:
- PID - Identificador único del proceso.
- UID - Identificador del usuario que lo ha creado.
- GID - Identificador del grupo/s al que pertenece el usuario que lo ha creado.
- PPID - Identificador del proceso padre.

Para poder obtener información sobre los procesos que actualmente se están ejecutando y monitorizar sus estados, podemos ayudarnos de los siguientes comandos.

ps [opciones]

Muestra la lista de procesos del sistema junto a sus características (PID, UID, GID, memoria, fecha, estado...). Las opciones que tenemos para su uso son:
-a - Muestra los procesos creados por cualquier usuario y asociados a un terminal.
-l - Formato detallado e indicando la prioridad.
-u - Incluye al usuario propietario y la hora de inicio.
-U - Listado de los procesos creados por el usuario que indiquemos.
-x - Muestra todos aquellos procesos que no están asociados con ningún terminal, como pueden ser los demonios.

pstree [opciones] [PID] [usuario]

Muestra la jerarquía de los procesos de nuestro sistema en una estructura de árbol. Podemos especificar el PID del proceso que queremos que parta el árbol o que muestre sólo aquellos pertenecientes a dicho usuario. Las opciones que tenemos para su uso son:
-a - Incluye la línea de comandos que se usó para iniciar el proceso.
-c - Deshabilita las réplicas de un mismo proceso.
-h - Remarca la posición del proceso actual.
-n - Ordena los procesos por el PID. Por defecto, se ordenan por el nombre.
-p - Incluye el PID de los procesos en el árbol.

top [opciones]

Muestra la lista de procesos del sistema actualizados constantemente junto a sus características (PID, UID, GID, memoria, fecha, estado...). Se trata de un comando que abre una pantalla nueva interactiva, de la cual es posible salir con la orden "q". Las opciones que tenemos para su uso son:
-i - Ignora los procesos inactivos.
-d - Tiempo de refresco de la pantalla en segundos.

htop [opciones]

Versión mejorada de top que permite un mayor número de interacciones con el sistema. Se trata también de un comando que abre una pantalla nueva interactiva, de la cual es posible salir con la orden "q". Las opciones que tenemos para su uso son:
-s - Ordena por columna.
-u - Muestra sólo los procesos de un usuario.
-p - Muestra sólo información de los procesos introducidos con su PID.

atop [opciones]

Versión mejorada de top que permite un mayor número de interacciones con el sistema, se encuentra a medio camino entre top y htop. Su principal desventaja con htop es que no permite la interacción con el ratón, pero sí con el teclado. Las opciones que tenemos para su uso son:
A - Ordena de más ocupado a menos en cuanto a servicios.
M - Ordena en función del consumo de memoria.
D - Ordena en función de los accesos a disco.
N - Ordena en función del uso de red.

iotop [opciones]

Comando que abre una aplicación en la terminal para poder revisar toda la información de entrada/salida que va generando el sistema. Es interactivo en cierta manera con el teclado, por lo que existen opciones que nos pueden ayudar para analizar los sucesos listados. Las opciones que tenemos para su uso son:
p PID - Indica con el PID qué proceso se quiere monitorizar.
u USER - Indica con el USER qué usuario se quiere monitorizar.
a - Datos acumulados desde que lanzamos el comando.
P - Muestra sólo los procesos y no los hilos.
k - Muestra las medidas correspondientes en kilobytes.
t - Añade el tiempo de captura de cada línea
D - Ordena en función de los accesos a disco.
N - Ordena en función del uso de red.

nmon

Se trata de una aplicación que nos permite personalizar la información a mostrar durante el monitoreo del sistema. Cuando es lanzado, se muestran varias opciones y en caso de querer ver sólo algo relacionado con algún elemento, pulsamos la tecla correspondiente. Algunas de esas opciones son:
m - Muestra la información relacionada con el consumo de memoria.
c - Muestra la información relacionada con el consumo de CPU.
d - Muestra la información relacionada con el uso de los discos duros.
n - Muestra la información relacionada con el uso de las redes.
o - Muestra los recursos e información detallada del sistema.

fuser

Muestra la lista de los procesos que se encuentran utilizando un archivo/directorio indicado. Las opciones que tenemos para su uso son:
c - Directorio actual.
r - Directorio raíz.
m - Fichero o directorio indicado seguido de esta opción.
. - Directorio en el que nos encontramos.

jobs

Muestra la lista de los procesos que se encuentran trabajando en la terminal que nos encontramos, ya sea en primer o segundo plano. Las opciones que tenemos para su uso son:
l - Muestra algo más de información del estado de dicho proceso.
p - Muestra solo el PID del proceso.

free

Muestra la cantidad de memoria libre y usada por el sistema. Este comando nos ofrece información de forma genérica, no relacionada con un proceso en cuestión. Las opciones que tenemos para su uso son:
h - Muestra las unidades más adecuadas para su lectura.
b - Muestra en bytes.
k - Muestra en KB.
m - Muestra en MB.
g - Muestra en GB.
l - Muestra un listado más detallado de la memoria.

pmap

Muestra la cantidad de memoria ocupada por un proceso en cuestión. Las opciones que tenemos para su uso son:
x - Información de forma extendida.
X - Información de forma detallada.
XX - Información de la forma más detallada posible.
p - Muestra el path completo.

vmstat

Muestra información referente a los procesos, memoria, paginación, sistemas de entrada/salida, discos y actividad de la CPU. Las opciones que tenemos para su uso son:
a - Muestra la memoria activa e inactiva.
f - Muestra el total de tareas creadas desde el arranque del sistema.
t - Incluye la marca de tiempo.
w - Versión ampliada de los datos mostrados.

Viendo las utilidades que podemos disfrutar con este último comando, ¿cómo podríamos sacar por pantalla una tabla con todas las acumulaciones y datos posibles?

meminfo

Se trata de un fichero de texto que se encuentra en el directorio proc y que va recogiendo información a tiempo real de la memoria del sistema. cat/ proc/meminfo

HERRAMIENTAS MULTIPLATAFORMA:

1. Glances

Glances es una herramienta multiplataforma que hace un análisis completo de los elementos que conforman nuestro sistema. La información se va mostrando dinámicamente en función del tipo de dispositivo que estemos usando y la monitorización se puede hacer de forma local o a través de un servidor. Se trata de una herramienta de código abierto que permite a los colaboradores poder añadir nuevas funcionalidades.

2. Netdata

Netdata es una herramienta multiplataforma que monitoriza en tiempo real el sistema en el que lo activemos. Es capaz de capturar trazas y métricas de diferentes hardware, aplicaciones y sistemas a nivel local, remoto o incluso en la nube. Sus numerosos modos de uso indican su envergadura y complejidad, por lo que su manual de usuario también es bastante amplio. Se trata de una herramienta de código abierto que permite a los colaboradores poder añadir nuevas funcionalidades.