Universidad Zaragoza

Departamento de Informática e Ingeniería de Sistemas
Universidad Zaragoza

ESCUELA UNIVERSITARIA POLITÉCNICA DE TERUEL

GRADO EN INGENIERÍA INFORMÁTICA

ADMINISTRACIÓN DE SISTEMAS

PRÁCTICA 2

SHELL SCRIPT

Sergio Albiol-Pérez
Curso 2023/24

1. Objetivos de la práctica

El objetivo básico de la presente práctica es que el alumno sea capaz de desarrollar una serie de scripts en "**Scripting**" de forma adquiriera las habilidades correspondientes en la Administración de Sistemas Operativos y Redes de Computadores.

2. Actividades de laboratorio.

Ejercicio 1. Analiza las siguientes órdenes y describe su uso, al igual que todos aquellos argumentos que consideres interesantes.

- sort
- split
- kill
- ps
- pstree
- renice
- uptime
- hostname
- shutdown
- uname

Ejercicio 2. Realizar un Shell Script que nos muestre a aquellos usuarios que hace más de un cierto número de días que no ingresan en el sistema. Se dispone para ello de los siguientes comandos: grep, lastlog.

Donde lastlog da un listado de todos los usuarios en el sistema. Ejemplo: lastlog -t$DIAS da un listado de quien ha ingresado en el sistema hace menos de $DIAS. DIAS es una variable inicializada al principio del script, para poder cambiarla fácilmente fijarla en 30 días. El formato de salida de lastlog es el siguiente:

Username	Port	From	Latest
root	pts/2	155.210.68.207	vie jun 18 11:04:08 +0200 2013

Ejercicio 3. Realizar un Shell Script que analice y cuente el número de ficheros de un directorio dado, además tendrá que cumplir una serie de requisitos:

1. **Recibirá como parámetro el directorio a analizar.**
2. **Únicamente tendrá un parámetro, en caso contrario deberá mostrar "Número de argumentos incorrecto" y deberá finalizar la ejecución.**
3. **Si el parámetro suministrado no es un directorio, deberá informar con el siguiente mensaje "<parámetro> no es un directorio" y deberá finalizar la ejecución.**
4. **El script mostrará el contenido del directorio actual y además el contenido de sus subdirectorios, contando el número de ficheros de cada tipo (fichero normal, directorio, enlace simbólico, tipo de dispositivo… y mostrar por la salida estándar el total de cada tipo, junto con su porcentaje.**

Un posible resultado sería:
Ejecutando "./analiza /dev":

Análisis de /dev
1458 ficheros encontrados:
* 710 ficheros normales (48%)
* 9 directorios (0%)
* 24 dispositivos de bloque (1%)
* 700 dispositivos de caracteres (48%)
* 2 pipes (0%)
* 2 sockets (0%)
* 11 enlaces simbólicos (0%)

Ejercicio 4. En un directorio tenemos un conjunto de ficheros de log que crecen de forma ilimitada (podéis usar los ficheros que encontrareis en /var/log). Realizar un script que recorra los ficheros * .log, y que, para aquellos de más de 1000 líneas deje sólo las últimas 500 y guarde el resto comprimidas, en un fichero de copia * .log.bak.gz ubicándolo en un directorio diferente.

- En este directorio deben guardarse, para cada fichero log, las últimas 2 copias de seguridad. Para eso es necesario ir rotando los nombres de las copias:

(últimas 500 líneas) -> * .log.bak1.gz -> * .log.bak2.gz -> eliminado

- El programa durante su ejecución irá mostrando los ficheros encontrados y señalará aquellos que cumplan los requisitos.

Ejercicio 5. Realizar un Shell Script denominado diffd con la siguiente sintaxis:

diffd [-r] directorio1 [directorio2]

Donde el script deberá buscar todos los nombres de fichero ubicados en directorio1 y directorio2, exceptuando los nombres de los directorios, y a continuación mostrar las diferencias, o sea aquellos ficheros que aparecen en uno de los directorios, pero no en el otro, indicando para cada uno de ellos en que directorio se encuentran.

Por otra parte, será necesario introducir una serie de parámetros: directorio1 y directorio2, siendo los directorios donde realizaremos la comparación, y como podemos observar en la sintaxis directorio2 es opcional, esto quiere decir que si el usuario no lo especifica el script comparará directorio1 con el directorio actual.

Finalmente, si añadimos la opción –r (opcional) el script tendrá que realizar la acción inversa, mostrando los nombres de aquellos ficheros que se encuentran en los dos directorios.

Ejercicio 6. Escribir un shell script que solicite al usuario introducir por teclado una palabra y la longitud de la palabra, y a continuación el script nos muestre información referente al número de vocales, número de consonantes y/o la existencia del mismo número de vocales y consonantes.

Ejercicio 7. El correo de los usuarios se guarda en el directorio /var/mail, en archivos con el nombre de login de cada usuario.
Se desea un script monimail.sh que monitoree estos archivos, realizando lo siguiente:

a) si el usuario fue borrado, pero quedó su archivo de correo, listar estos usuarios.
b) si el operador invocó el programa sin parámetro finalizará.
c) si el operador dio un parámetro no numérico, avisa el error y termina.
d) si el operador da como parámetro un número, listar los usuarios que hace más de este número de días que no lee su correo, es decir, que el archivo con su nombre en /var/mail no ha sido accedido.
Sugerencia: usar comando find.

Ejercicio 8. Realizar un Shell Script denominado nchmod que nos visualice por el código en decimal asociado a los permisos que le queramos dar a un fichero.

Ejemplo:

nchmod r-sr-xr-x el resultado que devolverá será 4555

El shell script tendrá que controlar el número de caracteres insertados por el usuario, mostrando un mensaje de error donde especifiquemos el uso

Uso: $0 permisos-fichero (ejemplo r-srwxr-- tiene que tener 9 caracteres)"

Ejercicio 9. Generar un script que nos permita generar ficheros tar, inicialmente el script nos mostrará un menú con las siguientes opciones:

a. Generacion fichero tar
b. Extracción fichero tar
c. Visualización de la información del fichero tar
d. Listado de todos los archivos incluidos en el fichero tar

Sintaxis: sh generatar.sh

Ejercicio 10. Realizar un Shell Script que genere cuentas de usuario.

Funcionamiento: ./genera_usuario login nombre_real grupo

- Login: el login del usuario a crear.
- Nombre_real: datos del usuario (nombre y apellidos).
- Grupo: únicamente podrá ser: contabilidad, finanzas, estadística.

Requisitos:

- El número de parámetros tiene que ser tres, en caso contrario tendrá que mostrar "Número de parámetros incorrecto" y deberá finalizar la ejecución.
- El primer parámetro deberá tener entre 3 y 8 caracteres, estando todos ellos en minúsculas, en caso contrario deberá mostrar "Login incorrecto: el login "login" deberá tener entre 3 y 8 caracteres" o bien "Login incorrecto: el login "login" deberá estar en minúsculas" y deberá finalizar la ejecución.
- Si el segundo parámetro contiene el carácter ":" deberá mostrar "el nombre real no puede contener el carácter :" y deberá finalizar la ejecución.
- Si el tercer parámetro no es ni contabilidad, ni finanzas, ni estadística, entonces deberá mostrar "grupo incorrecto: el grupo tiene que ser contabilidad, finanzas, estadística" y deberá finalizar la ejecución.
- Si el grupo seleccionado no se ha creado todavía, crear el primer GID que esté libre en el sistema [100 – 999], y si no existe ninguno libre, entonces deberá mostrar "No hay GID's libres" y deberá finalizar la ejecución.
- El directorio de inicio del usuario estará en /HOME y deberá llamarse igual que el login del usuario, si a la hora de crearse se produce un error, entonces deberá mostar por consola el error pertinente.
- El usuario usará /bin/bash como shell de inicio.
- El script preguntará al usuario la contraseña asociada, para ello usará la orden passwd. En el caso de que el usuario se equivoque al insertar la contraseña tres veces consecutivas, entonces el script deberá bloquear la cuenta de usuario.
- Los ficheros de personalización se tomarán de /etc/skel

Finalmente deberás comprobar que la cuenta ha sido creada correctamente.

Ejercicio 11. Realizar un Shell Script que nos genere contraseñas cifradas de 10 caracteres aleatorios, siendo dichos caracteres los que a continuación se describen dentro de un vector:

VECTOR="0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz"

Posteriormente realizar la compresión de un fichero de texto:
casa bbb zzzzz ff → casa 3b 5z 2f

Y a continuación realizar un cifrado simple de cada uno de los caracteres:

Carácter cifrado = (Código ASCII(carácter) + número constante) Módulo 256

¿Cómo se podría desencriptar?

Nota: se puede usar el valor devuelto por la variable ***RANDOM***