ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

Titulación:

INGENIERO TÉCNICO EN INFORMÁTICA DE GESTIÓN

Título del proyecto:

TÍTULO DEL PROYECTO FIN DE CARRERA

Fco. Javier Luquin Oroz

Cándido Aramburu Mayoz

Pamplona, 20 de Julio de 2010

Índice

Prefacio

1. Objetivos del proyecto

El objetivo último de este proyecto es automatizar, en la medida de lo posible, el proceso de instalación, control, creación y restauración de las copias de seguridad en una red local mediante el uso de Scripts y SSH.

2. Aplicaciones

A pesar de tener distintas aplicaciones prácticas y más funcionalidades, la realización de este proyecto ha sido enfocada a la restauración automática del sistema de los ordenadores del laboratorio.

Durante el curso académico, distintos estudiantes utilizan Linux en sus prácticas de laboratorio, es por ello que a final del año académico o justo antes de empezar el curso, es necesario reinstalar Linux en todos los ordenadores del laboratorio con el fin de que los estudiantes del nuevo curso académico encuentren un SO limpio.

Realizar la instalación de los 20 ordenadores uno por uno es una tarea muy repetitiva que conlleva demasiado tiempo. ¿Y si solo tuviésemos que ejecutar un script y esperar hasta que todos los sistemas hayan sido restaurados?

Los scripts utilizados en este proyecto nos permiten, entre otras cosas:

- Realizar la restauración del sistema de los 20 ordenadores del laboratorio.
- Crear las imágenes que se utilizaran para la restauración del sistema.
- Redimensionar las particiones de un ordenador destino cuya IP indiquemos.
- Restaurar el sistema del aquel ordenador cuya IP indiquemos.

3. Prerrequisitos

El sistema operativo utilizado para la implementación del proyecto es GNU/Linux, en concreto Ubuntu 9.10 y su derivado Xubuntu (similar pero con el entorno de escritorio Xfce en lugar de Gnome). Sin embargo, el uso de scripts facilita la portabilidad de los mismos siendo también ejecutables en cualquier distribución GNU/Linux (Suse, Debian, Kubuntu, Xubuntu…).

La ejecución de los Scripts, al estar escritos en bash, es posible con cualquier instalación estándar del sistema operativo. Sin embargo, para conectarse de manera

remota a los diferentes ordenadores de la red local los diferentes scripts usan SSH. Es por ello que será necesaria una única instalación, la del paquete cliente ssh (cuya instalación se explicara en el capitulo introductorio).

Dado que, como se explicara posteriormente, la restauración del sistema del ordenador remoto se hace mediante ssh, es requisito indispensable que el ordenador destino este encendido y funcionando en la partición correspondiente a Backup-ed (no hace falta llegar al entorno de escritorio, con dejarlos encendidos en la pantalla de acceso al sistema será suficiente).

Dado que ssh trabaja de manera remota y no visual, si un usuario se loguea físicamente en el ordenador destino no influye negativamente en la ejecución del script. Pues los procesos ejecutándose no se detendrán (a no ser que dicho usuario los pare premeditadamente con los comandos ps y kill). Sin embargo, el hecho de que alguien apague el ordenador destino mientras estamos efectuando la restauración remota si afectara a la realización de la misma.

Es requisito indispensable y obligatorio que el administrador de la red (en este caso el servicio informático de la universidad pública de navarra) no restrinja el acceso remoto a los ordenadores de la red local mediante SSH. Por defecto, la gran mayoría de las redes permiten el uso de SSH, véase como claro ejemplo internet. Sin embargo, puede que al extrapolar este proyecto a otras topologías de red con accesos más restringidos el usuario experimente problemas a la hora de ejecutar los diferentes Scripts (en cuyo caso deberá contactar con el administrador de la red).

Fruto de haber trabajado sobre Ubuntu 9.10 durante la creación de los Scripts, los mismos utilizan terminología acorde con dicho SO, es posible que la migración de los scripts a otro sistema operativo provoque conflictos a la hora de llamar a los scripts (el sistema no encontraría la ruta en donde están almacenados). Por ejemplo, que en SUSE, al escritorio se le llame Desktop en lugar de Escritorio (como ocurre en Ubuntu).

Para solucionar este problema, bastara con sustituir o modificar los nombres (en nuestro caso Escritorio por Desktop) en los diferentes Scripts. Tarea sencilla si abrimos el Script con un editor de texto y elegimos la opción reemplazar.

4. Recursos

- El laboratorio de electrónica digital del departamento de electrónica.
- Un ordenador portátil Sony vaio VGN-N38Z

5. Temas de la memoria

Capitulo Introductorio

En este capítulo se enseñara al lector cómo funciona el proceso de creación y restauración del sistema ideado y como instalar el paquete ssh client (necesario para la conexión remota mediante SSH). También se explicara cómo realizar una instalación básica de Ubuntu en dos particiones, requisito indispensable para la realización del proyecto, ya que una partición será la del acceso normal del usuario (edigital) y otra será a la que accederemos a la hora de hacer la restauración del sistema (backup-ed).

Capitulo 1: El Script Inicio

En este capítulo se explicara cómo actúa el script Inicio, así como la manera de iniciarlo, ya que este script es el único que el usuario deberá ejecutar. La ejecución de este script nos facilita y organiza el acceso a todos los demás de manera interactiva mediante el uso de un menú en Shell.

Capitulo 2: Los Scripts sshConexion, sshConexionBis y sus Scripts hijos

En el segundo capítulo describiremos cómo funcionan los scripts sshConexion y sshConexionBis. Scripts encargados de realizar la restauración del sistema en todos los ordenadores de la red local del laboratorio de electrónica Digital (19 ordenadores).

Capitulo 3: El Script sshConexionIndividual

En el tercer capítulo de esta memoria veremos cómo funciona el script sshConexionRemota. Similar a sshConexion y sshConexionBis pero en vez de ser un proceso automatizado para 19 IP's como los del capítulo anterior, añade la funcionalidad de poder elegir la IP del ordenador sobre el cual deseamos hacer la restauración del sistema.

Capitulo 4: Los Scripts backupLocal y backupRemoto

En este cuarto capítulo se explicaran las particularidades de los scripts backupLocal y backupRemoto. Encargados de crear las copias de seguridad de los ordenadores, el primero de ellos creara la copia del ordenador en el que ha sido ejecutado y el segundo hará lo propio, pero del ordenador sobre el cual hayamos introducido la IP.

Capitulo 5: El Script resizeRemoto

Aquí describiremos con detalle el script resizeRemoto, cuya funcionalidad es la de instalar gParted (herramienta grafica GNU para la manipulación de particiones) en un ordenador remoto (cuya IP deberemos especificar) y ejecutarlos de manera remota en nuestro ordenador, ofreciéndonos así, la posibilidad de manipular particiones de un ordenador al que no tenemos acceso físico.

Capitulo 6: El Script Permisos

En el capítulo final explicaremos como actúa este script, encargado de dar permisos a todos los scripts utilizados por el script Inicio.

Anexo 1: Comandos UNIX utilizados

En este primer anexo detallaremos en profundidad las características y funcionalidades de los diferentes comandos utilizados en los scripts.

Anexo 2: Particiones, características y tipos

La finalidad de este segundo anexo es introducir brevemente al lector inexperto en el tema de las particiones (características, tipos) y con la terminología utilizada en esta memoria.

Capitulo Introductorio

1. Breve introducción, ¿Qué es bash? ¿Qué es un script?

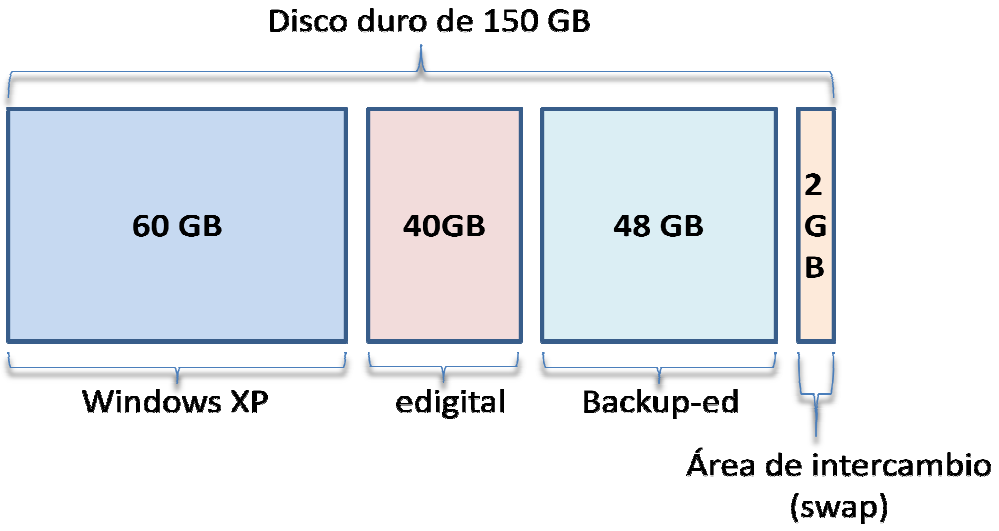
Un script (cuya traducción literal es guión) es un archivo de órdenes simple, que por lo regular se almacena en un archivo de texto plano. En nuestro caso, el archivo de texto termina en .sh para que el SO lo identifique como un ba**sh** script.

Bash es un intérprete de comandos de Linux. Está basado en el interprete Shell de Linux y es el intérprete de comandos por defecto en la mayoría de las distribuciones de Linux. Sera el lenguaje de programación que utilizaremos en nuestros scripts.

2. El proceso de restauración del sistema

El proceso de restauración (tarea más importante e idea fundamental del desarrollo del proyecto) se realiza de la siguiente manera:

Al ordenador del laboratorio se le instalan tres SO, de tal manera que tenga una partición para Windows y dos para Ubuntu. Una de ellas será de uso público (llamada edigital) para los estudiantes. Por otro lado, la segunda (llamada backup-ed) será a la que accederemos cuando deseemos restaurar el sistema (Ejemplo; a final de año deseamos formatear todas las particiones de Ubuntu edigital en la cual han trabajado los alumnos, y restaurar el sistema volviendo a tener todo como al principio del curso, es decir, una instalación limpia). Dado que la copia de edigital estará almacenada en backup-ed, dicha partición deberá poseer un tamaño suficiente para alojar su propio SO y la imagen de edigital.



Asimismo, el proceso de restauración puede ser de dos maneras:

- Almacenando la imagen de la partición de edigital localmente. Es decir, para cada ordenador, en la partición correspondiente a Backup-ed tendremos almacenada la imagen de la partición de edigital correspondiente a dicho ordenador. De esta opción se encargara el script sshConexion. Evidentemente, la partición de Backup-ed deberá tener suficiente espacio para almacenar la imagen de la partición edigital.

- Almacenando una imagen común en nuestro ordenador y pasándola por red a las diferentes particiones Backup-ed mediante scp. Para utilizar esta opción todas las particiones edigital de los ordenadores del laboratorio deberán tener el mismo tamaño que nuestra imagen. De esta otra opción se encargara el script sshConexionBis.

A parte de ello, y mediante el uso de scripts, también se añaden las siguientes funcionalidades:

- Restaurar a partir de una imagen previamente creada un ordenador ajeno al laboratorio.
- Redimensionar las particiones remotamente (para luego instalar el SO que deseemos).
- Crear las imágenes que se utilizaran para la restauración del sistema.

3. El proceso de instalación de Ubuntu 9.10

La instalación de Ubuntu no difiere en lo más mínimo de una instalación normal. En nuestro caso hemos usado los siguientes parámetros:

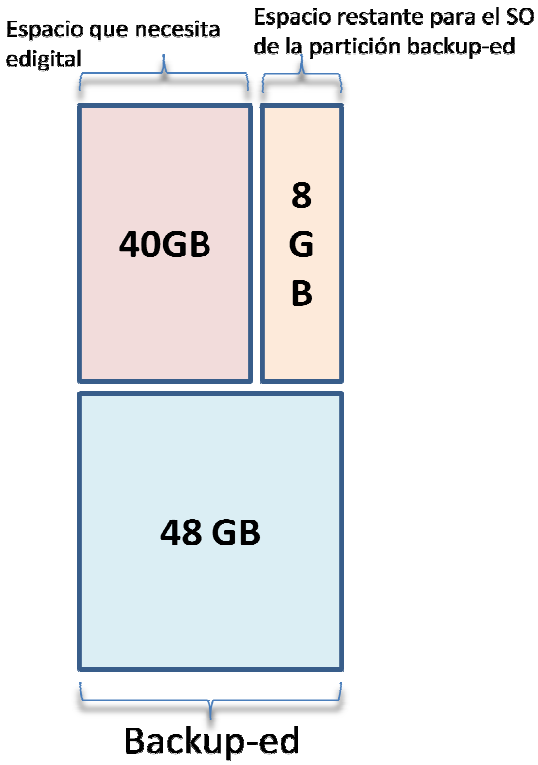
- Partición normal para uso de alumnos
  - Ubuntu 9.10
  - Nombre del sistema: edigital
  - Nombre de usuario : edigital
  - Contraseña: edigital

- Partición de recuperación
  - Xubuntu 9.10
  - Nombre del sistema: backup-ed
  - Nombre de usuario: backup-ed
  - Contraseña: caedigital

Como se ha indicado anteriormente, se ha de tener en cuenta que, en caso de que la partición backup-ed deba almacenar la imagen de la partición edigital, esta deberá tener un espacio suficientemente grande como para albergar la misma.

Ejemplo práctico:

- *Dado que una instalación básica de Xubuntu ocupa unos 2 GB la partición de recuperación deberá tener como mínimo 2GB más el espacio que le hayamos asignado a la partición de edigital.*
- *Si edigital ocupa 40000MB=40GB, entonces Backup-ed necesitara un mínimo de 2000MB+40000MB = 42000MB = 42GB mas el espacio libre que queramos darle (opcional, en este ejemplo es de 6 GB aunque con 1 GB habría sido suficiente)*



La elección del SO Xubuntu para la partición de Backup-ed es debida a razones de memoria. Básicamente, Xubuntu es igual que Ubuntu pero considerablemente más ligero al utilizar el entorno de escritorio Xfce en lugar de Gnome. No obstante, si el lector así lo desea, puede instalar cualquier otra distribución basada en Debian.

Al igual que todos los sistemas UNIX, la partición del área de intercambio (Swap) podrá ser compartida por ambos sistemas, por lo que no será necesaria la creación de dos particiones Swap (una para edigital y otra para backup-ed).

Si el lector nunca ha instalado Ubuntu o desea más información sobre el proceso de información se adjuntan los siguientes enlaces, en donde se detalla de manera precisa el proceso de instalación de Ubuntu 9.10:

http://paraisolinux.com/instalar-ubuntu-9-10-paso-a-paso/

http://blogubuntu.com/instalar-ubuntu-9-10/

4. Instalación de aplicaciones posteriores a la instalación del sistema en backup-ed

Una vez instalado Ubuntu, será necesario la instalación del paquete ssh (sudo apt-get install -y ssh). Este paquete nos permitirá acceder remotamente mediante ssh a dicho ordenador sin tener acceso físico al mismo. Lógicamente, solo será necesario en la partición de backup-ed (aquella a la cual deseamos tener acceso de manera remota)

```
edigital@edigital:~$ sudo apt-get install -y ssh
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
ssh ya está en su versión más reciente.
0 actualizados, 0 se instalarán, 0 para eliminar y 30 no actualizados.
```

5. Instalaciones de aplicaciones posteriores a la instalación del sistema en edigital (opcional, también puede hacerse una vez que la imagen haya sido restaurada)

Geda es un conjunto de aplicaciones de electrónica digital que los alumnos utilizan en las clases del laboratorio.

```
edigital@edigital:~$ sudo apt-get install –y geda
```

Capitulo 1: El Script inicio

1.1 Introducción

El script inicio será el único que el usuario deberá ejecutar. Los scripts que se utilizan están almacenados en una carpeta llamada ScriptsRedes y deberán estar situados en el escritorio.

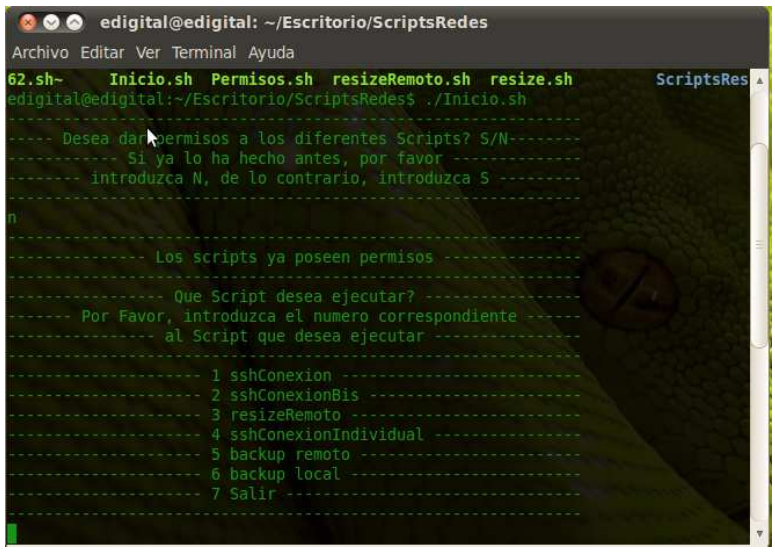
Abrimos una nueva sesión en la terminal y, usando el comando cd, nos movemos hasta el escritorio, donde estará situada la carpeta de los scripts y accedemos a la carpeta ScriptsRedes. Una vez allí. Ejecutamos el script Inicio.sh con el comando ./

```
edigital@edigital:~$ cd Escritorio/
edigital@edigital:~/Escritorio$ cd ScriptsRedes/
edigital@edigital:~/Escritorio/ScriptsRedes$
edigital@edigital:~/Escritorio/ScriptsRedes$ ./Inicio.sh
```

Si por algún casual el script no posee los permisos necesarios para ser ejecutado, introduciremos el siguiente comando por pantalla, situados en el directorio donde se encuentra el script inicio.sh

```
edigital@edigital:~/Escritorio/ScriptsRedes$ sudo chmod +x Inicio.sh
```

Una vez ejecutado, el script limpiara la pantalla del terminal (comando clear) y se nos preguntara si deseamos dar permisos de ejecución a todos los scripts que utilizaremos, si no lo hemos hecho, ejecutaremos esta opción ya que de lo contrario no podríamos ejecutarlos. Si por el contrario ya hemos ejecutado el script inicio anteriormente y les hemos dado permisos, diremos que no. Las respuestas son indiferentes de mayúsculas o minúsculas (s o S para sí, y n o N para no).



Tras responder, accederemos al menú del Script inicio. En donde, en función del script u opción que deseemos ejecutar, deberemos introducir un número u otro.

Como hemos visto en las imágenes, los scripts que podemos ejecutar aparecen numerados del 1 al 6:

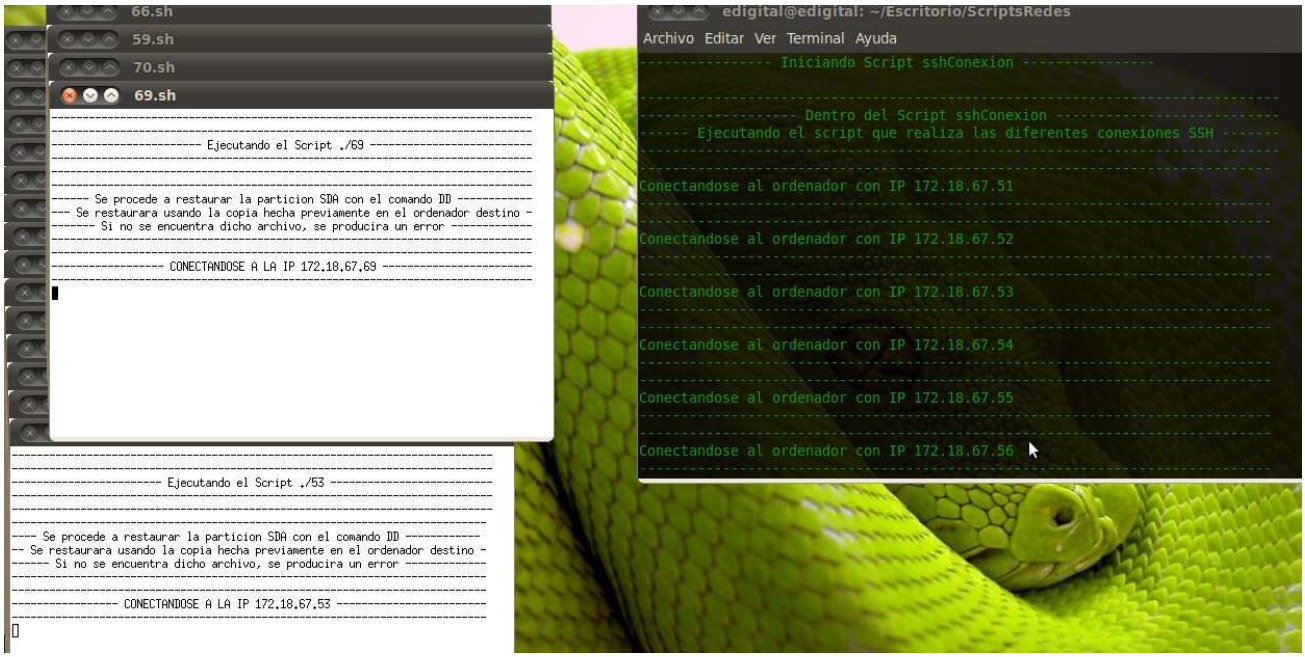
```
------------------------------------------------------------------
---------- 1 sshConexion ---------------------------
---------- 2 sshConexionBis -------------------------
---------- 3 resizeRemoto ---------------------------
---------- 4 sshConexionIndividual ------------------
---------- 5 backup remoto --------------------------
---------- 6 backup local ---------------------------
---------- 7 Salir ----------------------------------
------------------------------------------------------------------
```

A continuación, se procede a explicar las tareas que realizan los diferentes scripts. Su estructura y forma de actuar será descrita con más detalle en sus correspondientes capítulos.

1.2 Opción 1 sshConexion

Si el usuario introduce un 1, se ejecutara el script sshConexion. Este script realiza una restauración del sistema automática de los 19 ordenadores del laboratorio. Para ello, se conecta por ssh a cada uno de los ordenadores del laboratorio y abre dicha conexión en una nueva ventana, una por cada uno de los ordenadores.

Este script realiza la primera de las opciones de restauración explicadas en el capitulo introductorio (restauración del sistema utilizando la imagen previamente almacenada en el ordenador destino).



1.3 Opción 2 sshConexionBis

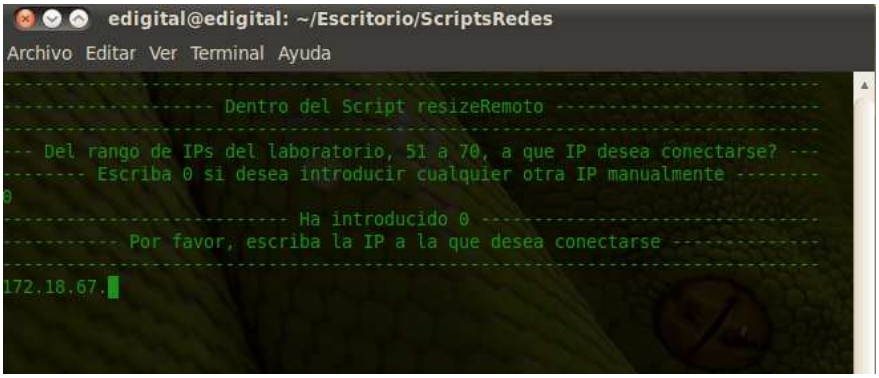
Si por el contrario el usuario introduce un 2, se ejecutara el script sshConexionBis, cuya funcionalidad es similar a la del script sshConexion.

Sin embargo, para el proceso de restauración, este script realiza la segunda de las opciones de restauración explicadas en el capitulo introductorio. La imagen a restaurar se transmitirá mediante scp al ordenador destino utilizando la imagen localizada en el ordenador desde el que se ejecuta el script.

1.4 Opción 3 resizeRemoto

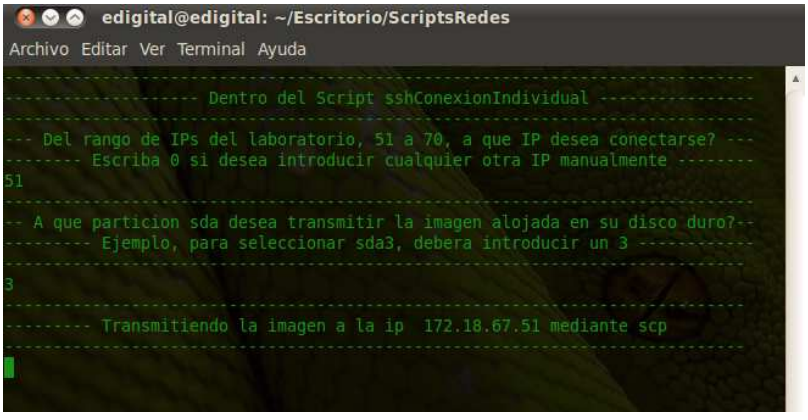
Este script nos pedirá el final del número de la IP de uno de los 19 ordenadores del laboratorio (del 51 al 70). También es posible introducir un 0 si queremos introducir cualquier otra IP a la que deseemos conectarnos.

Una vez conectados a la IP elegida, se conectara al ordenador destino mediante ssh, instalara el gestor de particiones Gparted y lo ejecuta en el ordenador destino. Una vez ejecutado nos los sacara por pantalla gráficamente en el ordenador desde el que lo hemos ejecutado. De esta manera, podremos editar las particiones en las que instalar Ubuntu (tanto la partición de edigital, como la de backup-ed) remotamente y de manera grafica.



1.5 Opción 4 sshConexionIndividual

La opción 4 (ejecución del script sshConexionIndividual) es similar a la del script sshConexionBis. Es decir, transmite la imagen desde el ordenador local al ordenador destino mediante scp y procede a restaurar el sistema. Sin embargo, a diferencia de la anterior este no es un proceso automatizado para todos los ordenadores del laboratorio. Este script añade la funcionalidad de elegir la IP del ordenador que queremos restaurar, así como la partición sda del ordenador destino en la cual queremos almacenar la imagen.



1.6 Opción 5 backup remoto

La función de este script es crear la imagen de la partición que seleccionemos (en nuestro caso la de edigital) en el ordenador cuya IP introduzcamos. Para ello, transmite el script backup (el mismo que utilizamos en la opción 6 backup-local) al ordenador destino mediante scp y lo ejecuta remotamente.

1.7 Opción 6 backup local

Este script crea la imagen de recuperación de la partición de edigital y la guarda en el ordenador en el cual ha sido ejecutado. Para ello, crea en el escritorio una carpeta llamada copiaseg donde almacenara la imagen de la partición de edigital (o de la partición que le indiquemos) bajo el nombre de recup. A continuación, saca por pantalla el conjunto de particiones que conforman el disco duro del ordenador en el que se ejecuta y pide al usuario que introduzca el número de la partición sda de la que desea hacer la copia.

1.8 Opción 7 salir

Se finaliza la ejecución del script Inicio y el usuario vuelve al prompt de Shell en el que se encontraba antes de ejecutar el Script Inicio.sh

1.9 El código del Script inicio.sh

```
clear
echo ------------------------------------------------------------
echo ----- Desea dar permisos a los diferentes Scripts? S/N--------
echo ------------ Si ya lo ha hecho antes, por favor -------------
echo -------- introduzca N, de lo contrario, introduzca S ---------
echo ------------------------------------------------------------
read LM
        if [ ``$LM'' = ``S'' ]; then
                echo ---------------------- Ha introducido S ------------------------
                echo ------------------ Iniciando Script Permisos -------------------
                ./Permisos.sh
        elif [ ``$LM'' = ``s'' ]; then
                echo ---------------------- Ha introducido s ------------------------
                echo ------------------ Iniciando Script Permisos -------------------
                ./Permisos.sh

        else
                echo ------------------------------------------------------------
                echo ---------------- Los scripts ya poseen permisos ----------------
        fi
echo ------------------------------------------------------------
echo ------------------ Que Script desea ejecutar? ------------------
echo ------- Por Favor, introduzca el número correspondiente ------
echo ---------------- al Script que desea ejecutar ----------------
echo ------------------------------------------------------------
echo ---------------------- 1 sshConexion -------------------------
echo ---------------------- 2 sshConexionBis ----------------------
echo ---------------------- 3 resizeRemoto ------------------------
echo ---------------------- 4 sshConexionIndividual ---------------
echo ---------------------- 5 backup remoto -----------------------
echo "---------------------- 6 backup local -------------------------"
echo ---------------------- 7 Salir -------------------------------
echo ------------------------------------------------------------
read LN
        if [ ``$LN'' = ``1'' ]; then
                echo ---------------- Iniciando Script sshConexion ----------------
                ./ScriptsRestaura/sshConexion.sh

        elif [ ``$LN'' = ``2'' ]; then
                echo -------------- Iniciando Script sshConexionBis ---------------
                ./ScriptsRestaura/sshConexionBis.sh

        elif [ ``$LN'' = ``3'' ]; then
                echo -------------- Iniciando Script resizeRemoto -----------------
                ./resizeRemoto.sh
        elif [ ``$LN'' = ``4'' ]; then
                echo -------------- Iniciando Script sshConexionIndividual --------
                ./sshConexionIndividual.sh
```

```
        elif [ ``$LN'' = ``5'' ]; then
                echo ---------- Iniciando Script backup remoto ----------
                clear
                echo ------------------------------------------------------------------------
                echo ---------- Dentro del Script Backup remoto ----------
                echo ------------------------------------------------------------------------
                echo ---------- A que IP desea conectarse para efectuar el backup? ----------
                echo ------------------------------------------------------------------------
                read y
                echo ------------------------------------------------------------------------
                echo "---------- Transmitiendo el Script backup mediante scp a la IP:$y"
                echo ------------------------------------------------------------------------
                scp /home/edigital/Escritorio/ScriptsRedes/backup.sh  backup-ed@$y:/home/backup-
ed/Desktop/backup.sh
                echo "---------- Procediendo a ejecutar el Script en la IP:$y"
                ssh -l backup-ed $y ./Desktop/backup.sh
        elif [ ``$LN'' = ``6'' ]; then
                echo "---------- Iniciando Script backup local ----------"
                ./backup.sh
        elif [ ``$LN'' = ``7'' ]; then
                clear
                echo ------------------------------------------------------------
                echo ---------- Saliendo.... ----------
                echo ---------- Ejecución del Script Inicio finalizada ----------
                echo ------------------------------------------------------------
                exit
        else
                clear
                echo ---------- Opción Incorrecta ----------
                echo -------- Por favor, ejecute de nuevo el Script Inicio --------
                echo ---------- E introduzca una opción válida ----------
                echo ------------------------------------------------------------
        fi
```

El objetivo de esta explicación no es el desarrollo de un manual completo del lenguaje de programación de scripts en bash. Sin embargo, con la intención de ilustrar al lector sobre el funcionamiento del script Inicio, se explicaran brevemente los comandos utilizados en este script. Tanto estos comandos como los que se explicaran brevemente en posteriores capítulos, están explicados con mayor detalle en el apartado “Anexo 1 comandos UNIX utilizados”.

echo nos sirve para hacer comentarios planos de texto y sacarlos por pantalla. Su utilización es muy simple, echo seguido del texto a mostrar. No obstante, cuando deseemos sacar una variable por pantalla (como por ejemplo la siguiente línea: `echo "---------- Transmitiendo el Script backup mediante scp a la IP:$y"`) deberemos poner comillas dobles.

read sirve para leer lo que el usuario introduzca por teclado (en nuestro caso la opción de menú elegida, la IP…). Se utiliza seguido del nombre que queremos dar a la variable en la cual almacenaremos los datos. En nuestro script en concreto y ordenados en orden de aparición:

- **Read LM** lo utilizamos para almacenar la respuesta del usuario a la pregunta sobre si desea dar permisos a los diferentes scripts.
- **Read LN** hace lo mismo pero para la opción del menú de scripts a ejecutar elegida.
- **Read y**, por último, es utilizada para almacenar la IP a la que queremos transmitir el script backup Remoto (opción 5).

Cuando utilicemos una de las variables en el script (LN, LM o y) deberán estar precedidas por el símbolo **$**, el cual indica al compilador de bash que se trata de una variable y debe leer su contenido. Por ejemplo, utilizando la variable y, en la cual hemos almacenada la IP, y mediante un hecho con comillas dobles, sacamos por pantalla la IP introducida por el usuario.

```
echo "---------- Transmitiendo el Script backup mediante scp a la IP:$y"
```

El funcionamiento del menú es simple. Se basa en una serie de clausulas if, else if utilizadas para evaluar la opción elegida por el usuario. En nuestro caso y a modo de ejemplo, las siguientes líneas:

```
if [ ``$LN'' = ``1'' ]; then
        echo ---------- Iniciando Script sshConexion ----------
        ./ScriptsRestaura/sshConexion.sh

elif [ ``$LN'' = ``2'' ]; then
        echo ---------- Iniciando Script sshConexionBis ----------
        ./ScriptsRestaura/sshConexionBis.sh
....
elif [ ``$LN'' = ``7'' ]; then
        clear
        echo ------------------------------------------------------------
        echo ------------------------ Saliendo.... ------------------------
        echo ---------- Ejecución del Script Inicio finalizada ------------
        echo ------------------------------------------------------------
        exit
else
        clear
        echo ------------------------------------------------------------
        echo ---------------------- Opción Incorrecta ----------------------
        echo -------- Por favor, ejecute de nuevo el Script Inicio --------
        echo --------------- E introduzca una opción válida ---------------
        echo ------------------------------------------------------------
fi
```

Son equivalentes a lo siguiente:

- Si el usuario ha introducido un 1 (correspondiente al script sshConexion) entonces:
    - Sacamos por pantalla “iniciando script sshConexion”
    - Ejecutamos el script sshConexión

- Si por el contrario lo que ha introducido el usuario es un 2 (correspondiente al script sshConexionBis):
    - Sacamos por pantalla “iniciando script sshConexionBis”
    - Ejecutamos el script sshConexionBis

  …

- Si por el contrario el usuario ha introducido un 7 (correspondiente a la opción salir) entonces:
    - Sacamos por pantalla “Saliendo”
    - Sacamos por pantalla “Ejecución del script Inicio finalizada”

- En cualquier otro caso (numero introducido distinto de 1,2,3,4,5,6,7)
    - Sacamos por pantalla “opción incorrecta”
    - Sacamos por pantalla “Por favor ejecute de nuevo el script inicio”
    - Sacamos por pantalla “E introduzca una opción válida”

Todas las opciones son prácticamente iguales, y usan el comando **./** para ejecutar el script que el usuario haya elegido.

./nombre del script a ejecutar.sh ejecuta los scripts que estén en el mismo directorio que el script inicio.

./directorio/nombre del script a ejecutar.sh ejecuta el script cuya ruta se indica en directorio (en nuestro caso sshConexion y sshConexionBis, que están dentro de la carpeta ScriptsRestaura, localizada dentro de la carpeta ScriptsRedes)

Todos los scripts que utilizamos están en la carpeta scriptsRedes situada en el Escritorio. Sin embargo, sshConexion y sshConexionBis que se encuentran dentro de una subcarpeta dentro de scriptsRedes llamada scriptsRestaura (en esta subcarpeta están todos los scripts que a su vez utilizan sshConexion y sshConexionBis, uno individualizado para cada IP del laboratorio, pero de ello ya hablaremos en siguiente capítulo). Se recomienda mantener esta distribución de los scripts con el fin de evitar tener que cambiar todos los directorios de ejecución.

Siguiendo con lo anteriormente expuesto, todas las opciones son prácticamente iguales, sacan algo por pantalla y ejecutan el script correspondiente a esa opción. Sin embargo, la opción 5 difiere del resto. Podría haberse creado un nuevo script backupRemoto, sin embargo, al ya tener creado el script Backup, la creación del script backupRemoto seria redundante. ¿Para qué queremos un nuevo script backupRemoto si con enviar por red el script backup y ejecutarlo remotamente es suficiente?

A continuación, procederemos a explicar el apartado 5:

```
elif [ ``$LN'' = ``5'' ]; then
        echo ---------- Iniciando Script backup remoto ----------
        clear
        echo ------------------------------------------------------------------------
        echo ---------- Dentro del Script Backup remoto ----------
        echo ------------------------------------------------------------------------
        echo ---------- A que IP desea conectarse para efectuar el backup? ----------
        echo ------------------------------------------------------------------------
        read y
        echo ------------------------------------------------------------------------
        echo "---------- Transmitiendo el Script backup mediante scp a la IP:$y"
        echo ------------------------------------------------------------------------
        scp /home/edigital/Escritorio/ScriptsRedes/backup.sh  backup-ed@$y:/home/backup-
ed/Desktop/backup.sh
        echo "---------- Procediendo a ejecutar el Script en la IP:$y"
        ssh -l backup-ed $y ./Desktop/backup.sh
```

Las explicaciones de los comandos echo, la clausula elif y read se omitirán al haber sido explicados previamente.

Los nuevos comandos en esta parte del código son clear, scp (para transmitir archivos por red) y ssh (para acceder remotamente).

clear limpia la pantalla de la terminal

scp

```
scp /home/edigital/Escritorio/ScriptsRedes/backup.sh  backup-ed@$y:/home/backup-ed/Desktop/backup.sh
```

Copia el script backup.sh (almacenado en la ruta /home/edigital/Escritorio/ScriptsRedes) al escritorio de la partición backup-ed del ordenador con IP $y (la que el usuario previamente ha introducido, véase read y).

Nótese que copiamos el script desde la ruta correspondiente a la organización por carpetas recomendada anteriormente. Si el lector posee el script backup en, por ejemplo una carpeta situada en el escritorio y llamada proyecto, la ruta debería ser /home/edigital/Escritorio/Proyecto/backup.sh.

De la misma manera, y como había sido comentado en el apartado “Prerrequisitos” nuestro ordenador se llama edigital, y el ordenador al que tenemos acceso, backup-ed. El ejemplo anteriormente expuesto, aplicado a un ordenador cuyo nombre fuese JosebaPC y estuviese basado en una distribución Linux como Xubuntu (que usa la nomenclatura Desktop en lugar de Escritorio) sería la siguiente /home/JosebaPC/Desktop/proyecto/backup.sh.

ssh

```
ssh -l backup-ed $y ./Desktop/backup.sh
```

Se conecta mediante ssh a la IP introducida previamente por el usuario (y almacenada en la variable y) y ejecuta el script backup.sh (que previamente hemos transmitido mediante scp y almacenado en el escritorio). El comando –l (de login) seguido de backup-ed le dice al sistema que se conecte a la IP usando dicho nombre de usuario (que en el caso de backup-ed es backup-ed).

Nótese que tras utilizar tanto el comando ssh como scp, el sistema nos pedirá que introduzcamos la contraseña de súper usuario del ordenador al que queremos acceder (en el caso de backup-ed es caedigital, para más información, véase de nuevo el capitulo introductorio).

Capitulo 2:

Los Scripts sshConexion, sshConexionBis y sus Scripts hijos

2.1 Introducción

La función de estos scripts es la realización de una restauración del sistema automática de los 19 ordenadores del laboratorio. Para ello, se conectan mediante ssh a cada uno de los ordenadores del laboratorio y abren dicha conexión en una nueva ventana, una por cada uno de los ordenadores. La principal diferencia entre ambos será, como veremos en el siguiente apartado, la metodología empleada a la hora de restaurar la imagen previamente creada en su partición correspondiente.

2.2 El código del script sshConexion

```
clear
echo ----------------------------------------------------------------------
echo ------------------- Dentro del Script sshConexion ---------------------------
echo ------ Ejecutando el script que realiza las diferentes conexiones SSH -------
echo ----------------------------------------------------------------------

echo  ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.51
echo  ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/ScriptsRestaura/51.sh &

echo  ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.52
echo  ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/52.sh &

echo  ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.53
echo  ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/53.sh &

echo  ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.54
echo  ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/54.sh &

echo  ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.55
echo  ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/55.sh &
```

```
echo ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.56
echo ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/56.sh &

echo ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.57
echo ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/57.sh &

echo ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.58
echo ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/58.sh &

echo ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.59
echo ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/59.sh &

echo ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.60
echo ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/60.sh &

echo ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.61
echo ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/61.sh &

echo ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.62
echo ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/62.sh &

echo ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.63
echo ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/63.sh &

echo ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.64
echo ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/64.sh &

echo ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.65
echo ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/65.sh &

echo ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.66
echo ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/66.sh &

echo ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.67
echo ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/67.sh &

echo ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.68
echo ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/68.sh &
```

```
echo  ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.69
echo  ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/69.sh &

echo  ----------------------------------------------------------------------
echo Conectándose al ordenador con IP 172.18.67.70
echo  ----------------------------------------------------------------------
xterm -e ./ScriptsRestaura/70.sh &

echo ------------ La ejecución del Script sshConexion ha finalizado ------------
echo ----------------------------------------------------------------------
```

Como es posible apreciar en el código fuente del script sshConexion, todas las llamadas a las diferentes IP's son prácticamente iguales:

- Sacamos por pantalla la IP a la que vamos a conectarnos.
- Abrimos una nueva terminal con el comando xterm –e (-e significa que queremos que se ejecute recurrentemente) y la ruta del script a ejecutar (uno diferente para cada IP).
- Hacemos lo mismo pero para la siguiente IP.

Hay 19 scripts diferentes, uno para cada IP. Su nomenclatura es muy sencilla, a la IP 172.18.67.64 le corresponderá el script 64.sh, a la IP 172.18.67.60 le corresponderá el script 64.sh, y así sucesivamente.

Todos los scripts de este capítulo (sshConexion, sshConexionBis y los scripts que utilizan) están localizados en una subcarpeta llamada scriptsRestaura dentro de la carpeta scriptsRedes.

La única diferencia entre los scripts sshConexion y sshConexionBis es su funcionalidad y el nombre de sus scripts hijos.

- sshConexion realiza restauración del sistema usando la imagen de la partición almacenada en el disco duro local del ordenador remoto. Sus scripts hijos tienen como nombre el ultimo numero de la IP a la que se conectan (Ejemplo 60.sh)

- sshConexionBis realiza la restauración del sistema usando la imagen alojada en el ordenador desde el que se ejecuta. Para ello, se transmite la imagen mediante scp usando la red local y luego se copia en la partición del ordenador destino. Cada uno de sus scripts hijos tienen como nombre el ultimo numero de la IP a la que se conectan unido al adjetivo bis (Ejemplo 60bis.sh)

Evidentemente, la única diferencia entre ambos scripts es la manera de llamar a sus scripts hijos, en sshConexion cada llamada acabara en “ultimo-nºIP.sh” mientras que en sshConexion cada llamada acabara de la forma “ultimo-nºbis.sh”.

De la misma manera, cada uno de los scripts individualizados (60.sh, 61.sh, 59bis.sh) tendrán la misma estructura. Sus posibles diferencias son las siguientes:

- Al estar individualizados para cada ordenador, uno podría restaurar la partición de de edigital en sda4 y otro en sda7, al ser donde respectivamente y para cada ordenador, esta almacenada la partición de edigital.
- Las rutas podrían ser diferentes (/home/edigital/escritorio o /home/edigital/Desktop).
- Evidentemente, la dirección IP de cada script será distinta.

2.3 Ejemplo de script hijo de sshConexion: 60.sh

```
echo ----------------------------------------------------------------------
echo ----------------------------------------------------------------------
echo ------------------------ Ejecutando el Script ./60 --------------------------
echo ----------------------------------------------------------------------
echo ----------------------------------------------------------------------
echo  ----------------------------------------------------------------------
echo  ------ Se procede a restaurar la partición SDA con el comando DD ------------
echo  --- Se restaurara usando la copia hecha previamente en el ordenador destino -
echo  ------- Si no se encuentra dicho archivo, se producirá un error -------------
echo  ----------------------------------------------------------------------
echo  ----------------------------------------------------------------------
echo  ------------------- CONECTANDOSE A LA IP 172.18.67.60 ------------------------
echo  ----------------------------------------------------------------------

ssh -l backup-ed 172.18.67.60 sudo dd if=/home/backup-ed/Escritorio/copiaseg/recup of=/dev/sda8

echo ----------------------------------------------------------------------
echo ------------------------ Finalizado Script ./60----------------------------
echo ----------------------------------------------------------------------
```

La sintaxis del comando ssh es la siguiente:

- La conexión por ssh va seguida de la opción –l (de login), la cual indica al sistema que el nombre de usuario con el cual conectarse al ordenador remoto es backup-ed.
- A continuación escribimos la IP a la que deseamos conectarnos y el comando a ejecutar.
- El comando usa la orden dd, la cual copia un input file (if) en donde le indiquemos en el apartado output file (of).

En if escribiremos la ruta en donde se encuentra la copia de seguridad de la partición (en nuestro caso la copia se llama recup y está dentro de la carpeta copiaseg, situada en el escritorio).

```
ssh -l backup-ed 172.18.67.60 sudo dd if=/home/backup-ed/Escritorio/copiaseg/recup of=/dev/sda8
```

En of pondremos la ruta donde queremos que se copie el archivo seleccionado anteriormente (recup). Que no es sino la partición sda donde estaba almacenada la partición de edigital, sobrescribiendo de esta manera la partición para almacenar en ella la imagen de la partición original.

2.4 Ejemplo de script hijo de sshConexionBis: 60bis.sh

Este script es prácticamente igual al anterior en todo lo que respecta al uso del comando ssh.

```
echo ----------------------------------------------------------------------
echo ----------------------------------------------------------------------
echo ---------------------- Ejecutando el Script ./60bis ------------------------
echo ----------------------------------------------------------------------
echo ----------------------------------------------------------------------
echo ----------------------------------------------------------------------
echo --------- Se procede a restaurar la partición SDA con el comando DD ---------
echo -----  Se restaurara pasando la imagen mediante ssh a través de la red ------
echo ----- Es posible que ocurran fallos durante la transferencia de archivos ----
echo ----------------------------------------------------------------------
echo ----------------------------------------------------------------------
echo ----------------- CONECTANDOSE A LA IP 172.18.67.60 -------------------------
echo ----------------------------------------------------------------------

scp /home/backup-ed/Desktop/copiaseg/recup  backup-ed@172.18.67.60:/home/backup-
ed/Escritorio/copiaseg/recup

echo ----------------------------------------------------------------------
echo ------------- Se ha finalizado de traspasar la imagen ----------------------
echo ---- Se procede a efectuar el backup en el ordenador remoto mediante ssh ----
echo ----------------------------------------------------------------------

ssh -l backup-ed 172.18.67.60 sudo dd if=/home/backup-ed/Escritorio/copiaseg/recup of=/dev/sda8

echo ----------------------------------------------------------------------
echo ------------------------- Finalizado Script ./60bis--------------------------
echo ----------------------------------------------------------------------
```

El comando scp (Secure Copy) es un medio de transferencia segura de archivos informáticos entre un ordenador local y otro remoto usando el protocolo ssh (secure shell).

En nuestro caso queremos copiar un archivo de nuestro ordenador al remoto por lo que primero escribimos la ruta en donde se encuentra el archivo a transmitir (en nuestro caso la imagen de la partición) y después donde copiarla.

Con la intención de tener una mejor organización de la partición de backup-ed, hemos almacenado la imagen de la partición de edigital en la carpeta copiaseg situada en el escritorio y le hemos llamado recup. De esta manera, en nuestro caso la ruta donde estará almacenada la imagen será /home/edigital/Escritorio/copiaseg/recup.

```
scp /home/backup-ed/Desktop/copiaseg/recup  backup-ed@172.18.67.60:/home/backup-ed/Escritorio/copiaseg/recup
```

El lugar donde queremos almacenar la copia en el ordenador destino puede ser cualquiera, con la condición de que luego especifiquemos la misma ruta en el comando dd (cuyas particularidades se han descrito en el apartado anterior).En nuestro caso, para no tener que cambiar nada del comando dd, la hemos almacenado en /home/backup-ed/Escritorio/copiaseg con el nombre de recup.

Como podemos observar en el script, antes de la ruta en donde guardar el archivo a copiar deberemos especificar el nombre de usuario con el que conectarnos al ordenador remoto seguido de un símbolo arroba (@) y la IP a la que deseamos conectarnos.

Tras la lectura de esta explicación, probablemente el lector se pregunte cual es la necesidad de efectuar dos comandos para restaurar la imagen del sistema ¿Por qué no copiar directamente a la partición sda correspondiente utilizando el comando scp? Esta opción ha sido descartada al existir la posibilidad de que ocurra un fallo durante la transferencia de archivos. Si esto ocurre y estamos copiando directamente sobre la partición sda posiblemente dejemos dicha partición inservible o con gran cantidad de archivos corruptos.

Por el contrario, si hay fallos al copiar la imagen en la carpeta copiaseg del ordenador destino, la ejecución del script se detendrá automáticamente (al haber lanzado el comando dd un mensaje error) y simplemente tendremos que volver a ejecutarlo para que el script sobrescriba la imagen defectuosa por una nueva.

Capitulo 3:

El script sshConexionIndividual

3.1 Introducción

La funcionalidad de este script es efectuar una restauración del sistema en la partición sda que elijamos de aquel ordenador cuya IP indiquemos.

Es especialmente útil si lo usamos junto al script resizeRemoto (capitulo 5). Por ejemplo, y de manera esquemática:

1. Tenemos una imagen de una partición de 12GB almacenada en nuestro ordenador,
2. Ejecutamos el script resizeRemoto
3. Usando el script resizeRemoto hacemos una partición de 12GB ext3 en un ordenador cuya IP es 172.18.67.67.
4. A continuación ejecutamos el script sshConexionIndividual.
5. Introducimos la IP 172.18.67.67 y introducimos un 3 (correspondiente a la partición sda3)
6. Una vez finalizado el script sshConexionIndividual tendremos la imagen de 12GB cargada en la partición sda3 del ordenador destino.

Nótese que para que este proceso no de ningún error han de cumplirse dos requisitos:

- El tamaño de la imagen y la partición en la que la copiamos deben ser exactamente iguales, en caso de ser diferentes, el tamaño de la partición sda en la que almacenaremos la imagen será mayor que el tamaño de la imagen, nunca al revés.
- Para que la imagen cargada en el ordenador destino funcione correctamente los dos ordenadores deberán tener un hardware similar (como ocurre en el laboratorio).

Evidentemente, el uso de este script no acarrea la ejecución obligatoria del script resizeRemoto. También puede usarse para restaurar solo un ordenador en concreto del laboratorio.

3.2 El código del script sshConexionIndividual

```
clear
echo ------------------------------------------------------------------------
echo -------------------- Dentro del Script sshConexionIndividual ----------------
echo ------------------------------------------------------------------------
echo --- Del rango de IPs del laboratorio, 51 a 70, a que IP desea conectarse? ---
echo -------- Escriba 0 si desea introducir cualquier otra IP manualmente --------
read x
if [ ``$x'' = ``0'' ]; then
        echo --------------------------- Ha introducido 0 -------------------------------
        echo ----------- Por favor, escriba la IP a la que desea conectarse --------------
        echo ------------------------------------------------------------------------
        read y
        echo ------------------------------------------------------------------------
        echo "-- A que partición sda desea transmitir la imagen alojada en su disco duro?--"
        echo --------- Ejemplo, para seleccionar sda3, deberá introducir un 3 ------------
        echo  ------------------------------------------------------------------------
        read l
        echo  "--------- Transmitiendo la imagen a la ip  $y mediante scp"

        scp /home/edigital/Escritorio/copiaseg/recup  backup-ed@$y:/home/backup-
ed/Desktop/copiaseg/recup

        echo  ------------------------------------------------------------------------
        echo ------------- Se ha finalizado de traspasar la imagen -----------------------
        echo ---- Se procede a efectuar el backup en el ordenador remoto mediante ssh ----
        echo "----------------- Utilizando para ello la partición sda$l --------------------"
        echo  ------------------------------------------------------------------------

        ssh -l backup-ed $y sudo dd if=/home/edigital/Escritorio/copiaseg/recup of=/dev/sda$l
else
        echo ------------------------------------------------------------------------
        echo "-- A que partición sda desea transmitir la imagen alojada en su disco duro?--"
        echo --------- Ejemplo, para seleccionar sda3, deberá introducir un 3 ------------
        echo  ------------------------------------------------------------------------
        read t
        echo  ------------------------------------------------------------------------
        echo  "--------- Transmitiendo la imagen a la ip  172.18.67.$x mediante scp"
        echo  ------------------------------------------------------------------------

        scp /home/edigital/Escritorio/copiaseg/recup  backup-ed@172.18.67.$x:/home/backup-
ed/Desktop/copiaseg/recup

        echo ------------------------------------------------------------------------
        echo ---------------- Se ha finalizado de traspasar la imagen --------------------
        echo --------- Se procede a ejecutarlo en el ordenador remoto mediante ssh -------
        echo "----------------- Utilizando para ello la partición sda$t --------------------"
        echo ------------------------------------------------------------------------
        ssh -l backup-ed 172.18.67.$x sudo dd if=/home/edigital/Escritorio/copiaseg/recup of=/dev/sda$t
fi
echo ------------------------------------------------------------------------
echo -------------------- Finalizado Script sshConexionIndividual ----------------
echo ------------------------------------------------------------------------
```

Al ejecutar este script se nos pedirá que introduzcamos el numero en el que acaba la IP de cualquiera d los ordenadores del laboratorio o un 0 si deseamos introducir la IP entera manualmente.

Para realizar esto hemos estructurado en script usando una clausula if else.

- Pedimos el ultimo numero de la IP y lo almacenamos en la variable x
- Si lo que nos ha introducido el usuario es un 0 (x=0)
  - Pedimos la IP entera y la almacenamos en la variable y
  - Pedimos al usuario que nos introduzca el número de la partición sda en la que desea alojar la imagen y lo almacenamos en la variable l.
  - Transmitimos mediante scp la imagen al ordenador con IP y
  - Copiamos la imagen a la partición sda l usando ssh y el comando dd

- Si lo que nos ha introducido no es un 0 (x≠0)
  - Pedimos al usuario que nos introduzca el número de la partición sda en la que desea alojar la imagen y lo almacenamos en la variable t.
  - Transmitimos mediante scp la imagen al ordenador con IP 172.18.67.x
  - Copiamos la imagen a la partición sda t usando ssh y el comando dd

La estructura de los comandos scp y ssh usados aquí es similar a la de los scripts hijos de sshConexion y sshConexionBis. Sin embargo, en este script hemos utilizado las variables x, y, l, t como parte de los comandos.

Como se ha explicado en el Capitulo 1, para que el sistema sepa que debe leer el contenido de una variable, esta deberá estar precedida por el símbolo $.

De esta manera, en el supuesto de que hayamos introducido un 0 (escribir la IP manualmente), a continuación la IP 172.18.67.65 y por ultimo un 3 (equivalente a sda3) la línea:

```
ssh -l backup-ed $y sudo dd if=/home/edigital/Escritorio/copiaseg/recup of=/dev/sda$l
```

Equivale a lo siguiente:

```
ssh -l backup-ed 172.18.67.65 sudo dd if=/home/edigital/Escritorio/copiaseg/recup of=/dev/sda3
```

Otro ejemplo, hemos introducido 61, y a continuación un 4 (equivalente a sda4) la línea:

```
ssh -l backup-ed 172.18.67.$x sudo dd if=/home/edigital/Escritorio/copiaseg/recup of=/dev/sda$t
```

Equivale a lo siguiente:

```
ssh -l backup-ed 172.18.67.61 sudo dd if=/home/edigital/Escritorio/copiaseg/recup of=/dev/sda4
```

Capitulo 4:

Los scripts backupLocal y backupRemoto

4.1 Introducción

A pesar de la terminología utilizada en el script Inicio.sh y por consiguiente, en el titulo de este capítulo los scripts backupLocal y backupRemoto no existen como tal. Solo existe un único script backup. La opción 6 (backupLocal) simplemente ejecuta el script backup.

Por otro lado, la opción 5 (backupRemoto) transmite mediante scp el script backup a la ip que el usuario haya indicado y lo ejecuta remotamente.

Podría haberse creado un nuevo script backupRemoto, sin embargo, al ya tener creado el script Backup, la creación del script backupRemoto seria redundante. ¿Para qué queremos un nuevo script backupRemoto si con enviar por red el script backup y ejecutarlo remotamente es suficiente? La parte correspondiente al “pseudo script” backupRemoto están explicados (al ser el mismo parte del script Inicio.sh) en la página 20 (capitulo 1: El Script Inicio).

La funcionalidad de este script es la creación de la copia de seguridad de una partición (a seleccionar por el usuario). Una vez ejecutado, este script saca por pantalla todas las particiones que conforman el disco duro en el que ha sido ejecutado y pregunta al usuario sobre que partición debe hacer la copia de seguridad. Para realizar la imagen de la partición utiliza el comando dd.

Como se explica en el apartado “El código del script backup” y al igual que en los capítulos anteriores. Almacenaremos la imagen de la partición (bajo el nombre de recup) en una carpeta llamada copiaseg situada en el Escritorio de la partición backup-ed. No hará falta que dicha carpeta este creada, pues el propio script se encargara de ello.

Es importante recordar que en el resto de scripts seguimos este supuesto, pues al realizar la restauración del sistema utilizando el comando dd (sshConexion, sshConexionBis, sshConexionIndividual) suponemos que la localización de la imagen a restaurar sigue dicho supuesto (home/backup-ed/Escritorio/copiaseg/recup).

4.2 El código del script backup

```
sudo mkdir -m +x ~/Escritorio/copiaseg
clear
echo ------------------------------------------------------------------------
echo ----------------------- Dentro del Script Backup --------------------------
echo ------------------------------------------------------------------------
sudo fdisk -l | grep sda
echo ------------------------------------------------------------------------
echo ---------------- De que partición sda desea hacer el Backup? -----------------
echo --------- Ejemplo, para seleccionar sda3, deberá introducir un 3 ------------
echo ------------------------------------------------------------------------
read l
echo ------------------------------------------------------------------------
echo "---------- Procediendo a realizar el backup de la partición sda$l ----------"
echo "------- mediante el comando dd, este proceso durara unos 25 minutos ---------"
echo ------------------------------------------------------------------------

sudo dd if=/dev/sda$l of=/home/edigital/Escritorio/copiaseg/recup

echo ------------------------------------------------------------------------
echo --- Proceso finalizado, la imagen ha sido guardada en /Escritorio/copiaseg --
echo ------------------------------------------------------------------------
echo ------------------------ Finalizado Script backup --------------------------
echo ------------------------------------------------------------------------
```

Una vez ejecutado, este script realiza las siguientes acciones:

1. Crea la carpeta copiaseg en el escritorio con el comando:

```
sudo mkdir -m +x ~/Escritorio/copiaseg
```

Mkdir crea un directorio (carpeta). La opción –m le indica al sistema que cree dicha carpeta con los privilegios indicados a continuación (en este caso +x, es decir todo el mundo tiene permisos para leer y escribir en dicha carpeta).

En el caso de que el script haya sido ejecutado remotamente estaremos en el directorio home. El símbolo ~ precediendo al directorio /Escritorio/copiaseg hace que el interprete cree la carpeta en el directorio indicado sin tener en cuenta en que directorio se encuentra actualmente.

2. Se sacan por pantalla las particiones que conforman el disco duro con el comando:

```
sudo fdisk -l | grep sda
```

Fdisk permite dividir en forma lógica un disco duro, sin embargo, para nuestro script solo nos interesara el uso de la opción –l , que permite listar las particiones, sectores y diversa información sobre el disco duro. Dada la extensión de la información que devolvía este comando, se ha utilizado el símbolo | seguido del comando grep para filtrar el flujo de datos resultante de la ejecución del comando fdisk-l.

Grep es una utilidad de la línea de comandos UNIX. Su funcionamiento es el siguiente:

1) Toma una expresión regular de la línea de comandos.
2) Lee la entrada estándar o una lista de archivos.
3) Imprime las líneas que contengan coincidencias para la expresión regular.

En nuestro caso, el flujo de datos entrante será el resultado de la ejecución del comando fdisk –l. Y la expresión regular sobre la cual buscar coincidencias será sda. De esta manera, solo se nos listara por pantalla la información correspondiente a las particiones sda.

3. Se pide al usuario que introduzca el numero de la partición sda de la cual se desea hacer la imagen y lo almacena en la variable l

4. Se inicia el proceso de creación de la imagen con el comando:

```
sudo dd if=/dev/sda$l of=/home/edigital/Escritorio/copiaseg/recup
```

Las particularidades del comando dd ya han sido explicadas en capítulos anteriores. Sin embargo si en el proceso de restauración del sistema copiábamos la imagen desde la carpeta copiaseg hasta la partición sda, a la hora de crear la imagen haremos justo lo contrario. Es decir, copiaremos desde la partición sda a la carpeta copiaseg previamente creada.

En este caso, copiaremos bajo el nombre de recup el contenido integro de la partición sda cuyo número haya indicado el usuario (almacenado en la variable $l) y lo guardaremos en la carpeta copiaseg, ubicada en el Escritorio.

Tras copiar la imagen de la partición sda, la ejecución del script backup finaliza.

Capitulo 5:

El script resizeRemoto

5.1 Introducción

Este sencillo script nos permite gestionar las particiones de un ordenador de manera visual sin tener acceso físico al mismo. Para ello utiliza ssh y Gparted, un programa utilizado para crear, eliminar, redimensionar, inspeccionar y copiar particiones o sistemas de archivos.

Gparted utiliza la biblioteca libparted para detectar y manipular dispositivos y tablas de partición. Esta programado en C++ y se distribuye bajo la licencia libre de GNU. El programa, manuales y diversa información pueden ser encontrados en la página web oficial de Gparted (http://gparted.sourceforge.net/).

5.2 El código del script resizeRemoto

```
clear
echo ------------------------------------------------------------------------
echo ------------------- Dentro del Script resizeRemoto -------------------------
echo ------------------------------------------------------------------------
echo --- Del rango de IPs del laboratorio, 51 a 70, a que IP desea conectarse? ---
echo -------- Escriba 0 si desea introducir cualquier otra IP manualmente --------
read x
if [ ``$x'' = ``0'' ]; then
        echo ---------------------------- Ha introducido 0 --------------------------------
        echo ----------- Por favor, escriba la IP a la que desea conectarse --------------
        echo ------------------------------------------------------------------------
        read y

        ssh -l backup-ed $y sudo apt-get install -y gparted
        ssh -X -l backup-ed $y sudo gparted

else
        ssh -l backup-ed 172.18.67.$x sudo apt-get install -y gparted
        ssh -X -l backup-ed 172.18.67.$x sudo gparted
fi
echo ------------------------------------------------------------------------
echo ------------------- Finalizado Script resizeRemoto -------------------------
echo ------------------------------------------------------------------------
```

La estructura de este script es muy sencilla. Al igual que en casos anteriores, el usuario tendrá dos maneras de elegir la IP destino organizado alrededor de una estructura if else:

- Introduciendo el último número de una de las IP's del laboratorio (Ejemplo: 60 para la IP 172.18.67.60). Se almacenara en la variable x

- Introduciendo un 0 para posteriormente introducir la IP manualmente. Se almacenara en la variable y

Una vez que el usuario ha introducido la IP nos conectamos al ordenador destino utilizando ssh e instalamos gparted. Si Gparted ya está instalado en el ordenador destino, el comando apt-get nos informara de que el programa ya está instalado y el script continuara con su ejecución normal.

```
ssh -l backup-ed $y sudo apt-get install -y gparted
```

El comando arriba citado se corresponde con la opción de introducir el número de IP manualmente. Almacenándose la misma en la variable y. Nos conectamos a la IP $y con nombre de usuario backup-ed y ejecutamos el comando "sudo apt-get install –y gparted". La opción –y (yes) le indica al intérprete de comandos que las respuestas a todas las preguntas/confirmaciones que surjan durante la instalación del paquete siempre serán "si".

A continuación, el programa ejecuta una sesión grafica remota (opción -X) en el ordenador destino. Se conecta utilizando backup-ed como nombre de usuario y ejecuta el programa gparted (comando "sudo gparted").

```
ssh -X -l backup-ed $y sudo gparted
```

Capitulo 6:

El script permisos

6.1 Introducción

La funcionalidad de este script es dotar de derechos de ejecución a todos los scripts que utiliza el script Inicio. Al ejecutar el script inicio, este preguntara al usuario si desea dar permisos de ejecución a los diferentes scripts. Si el usuario introduce s ó S (equivalentes a "si") el script inicio ejecutara permisos.sh.

6.2 El código del script permisos

```
clear
echo ----------------------------------------------------------------------------
echo ---------------------- Dentro del Script Permisos -------------------------
echo ----------------------------------------------------------------------------

sudo chmod +x ScriptsRestaura/51.sh
sudo chmod +x ScriptsRestaura/52.sh
sudo chmod +x ScriptsRestaura/53.sh
sudo chmod +x ScriptsRestaura/54.sh
sudo chmod +x ScriptsRestaura/55.sh
sudo chmod +x ScriptsRestaura/56.sh
sudo chmod +x ScriptsRestaura/57.sh
sudo chmod +x ScriptsRestaura/58.sh
sudo chmod +x ScriptsRestaura/59.sh
sudo chmod +x ScriptsRestaura/60.sh
sudo chmod +x ScriptsRestaura/61.sh
sudo chmod +x ScriptsRestaura/62.sh
sudo chmod +x ScriptsRestaura/63.sh
sudo chmod +x ScriptsRestaura/64.sh
sudo chmod +x ScriptsRestaura/65.sh
sudo chmod +x ScriptsRestaura/66.sh
sudo chmod +x ScriptsRestaura/67.sh
sudo chmod +x ScriptsRestaura/68.sh
sudo chmod +x ScriptsRestaura/69.sh
sudo chmod +x ScriptsRestaura/70.sh
sudo chmod +x ScriptsRestaura/51bis.sh
sudo chmod +x ScriptsRestaura/52bis.sh
sudo chmod +x ScriptsRestaura/53bis.sh
sudo chmod +x ScriptsRestaura/54bis.sh
sudo chmod +x ScriptsRestaura/55bis.sh
sudo chmod +x ScriptsRestaura/56bis.sh
sudo chmod +x ScriptsRestaura/57bis.sh
sudo chmod +x ScriptsRestaura/58bis.sh
sudo chmod +x ScriptsRestaura/59bis.sh
```

```
sudo chmod +x ScriptsRestaura/60bis.sh
sudo chmod +x ScriptsRestaura/61bis.sh
sudo chmod +x ScriptsRestaura/62bis.sh
sudo chmod +x ScriptsRestaura/63bis.sh
sudo chmod +x ScriptsRestaura/64bis.sh
sudo chmod +x ScriptsRestaura/65bis.sh
sudo chmod +x ScriptsRestaura/66bis.sh
sudo chmod +x ScriptsRestaura/67bis.sh
sudo chmod +x ScriptsRestaura/68bis.sh
sudo chmod +x ScriptsRestaura/69bis.sh
sudo chmod +x ScriptsRestaura/70bis.sh

sudo chmod +x ScriptsRestaura/sshConexion.sh
sudo chmod +x ScriptsRestaura/sshConexionBis.sh
sudo chmod +x resize.sh
sudo chmod +x resizeRemoto.sh
sudo chmod +x sshConexionIndividual.sh
sudo chmod +x backup.sh
sudo chmod +x Inicio.sh

echo ----------------------------------------------------------------------------
echo ------------ La ejecución del Script Permisos ha finalizado ----------------
echo ----------------------------------------------------------------------------
```

chmod (change mode) es un comando que nos permite dar privilegios de lectura, escritura, ejecución y acceso en UNIX. La opción +x significa que damos privilegios de ejecución al documento indicado seguidamente. Como bien puede observarse en el script, todas las llamadas son prácticamente iguales.

Siguiendo con lo expuesto en anteriores capítulos, se recuerda al lector que hemos supuesto una ordenación de los diferentes scripts por carpetas. De esta manera, estarán casi todos almacenados dentro de la carpeta ScriptsRedes, directorio en el que ya nos hemos situado para ejecutar Inicio.sh, por lo que no será necesario especificar ninguna ruta.

```
sudo chmod +x backup.sh
```

Mientras que sshConexion, sshConexionBis y sus scripts hijos estarán localizados en una subcarpeta llamada ScriptsRestaura dentro de la carpeta ScriptsRedes.

```
sudo chmod +x ScriptsRestaura/sshConexionBis.sh
```

Anexo 1:

Comandos UNIX utilizados

1. Clear

Clear es un comando estándar de todos los Sistemas Operativos basados en Unix. –Su función es limpiar los datos de la pantalla del terminal.

Dependiendo del sistema, clear usa la base de datos terminfo o termcap para deducir como limpiar la pantalla.

2. Echo

Echo es un comando para la impresión de un texto en pantalla. Es utilizado en las terminales de los sistemas operativos como Unix, GNU/Linux, o MS-DOS; dentro de pequeños programas llamados scripts o en lenguajes de más alto nivel avanzados como PHP.

3. Read

Read es un comando que nos permite leer los valores de la entrada estándar y asignarlos a variables.

4. Ssh

SSH (Secure SHell, en español: intérprete de órdenes segura) es el nombre de un protocolo y del programa que lo implementa, y sirve para acceder a máquinas remotas a través de una red. Permite manejar por completo la computadora mediante un intérprete de comandos, y también puede redirigir el tráfico de X para poder ejecutar programas gráficos si tenemos un Servidor X corriendo.

Además de la conexión a otros dispositivos, SSH nos permite copiar datos de forma segura (tanto ficheros sueltos como simular sesiones FTP cifradas), gestionar claves RSA para no escribir claves al conectar a los dispositivos y pasar los datos de cualquier otra aplicación por un canal seguro tunelizado mediante SSH.

5. Scp

Secure Copy o SCP es un medio de transferencia segura de archivos informáticos entre un host local y otro remoto o entre dos hosts remotos, usando el protocolo Secure Shell (SSH).

El cliente SCP más ampliamente usado es el programa scp del Intérprete de comandos, que es incorporado en la mayoría de las implementaciones de SSH.

6. Xterm -e

Xterm es un emulador de terminal para el sistema de ventanas X Window System aunque su origen es anterior a este. Fue desarrollado originalmente como un emulador de terminal independiente para la VAXStation 100 (VS100) por Mark Vandevoorde.

De las muchas opciones que posee este programa, particularmente útil para la realización de este proyecto ha sido la opción:

-e program [arguments...] Esta opción especifica el programa (y sus comandos) a ser ejecutado en la nueva ventana xterm. También establece el titulo de la ventana y debe ser la última de las opciones (en el caso de que hubiera más) en ejecutarse.

7. Sudo

El programa sudo (de las siglas en inglés de superuser -o substitute user- do) es una utilidad de los sistemas operativos basados en Unix que permite a los usuarios ejecutar programas con los privilegios de seguridad de otro usuario (normalmente el usuario root) de manera segura. Se instala por defecto en /usr/bin.

8. Fdisk –l

Fdisk es un programa de computadora disponible en varios sistemas operativos, el cual permite dividir en forma lógica un disco duro, siendo denominado este nuevo espacio como partición.

De las muchas posibilidades que nos brindan sus opciones, para la realización de este proyecto ha sido útil su opción –l (del ingles list).

-l Lista las tablas de partición de /dev/hd[a-d], /dev/sd[a-h] y /dev/ed[a-d].

9. Grep

Grep es una utilidad de la línea de comandos escrita para ser usada con el sistema operativo Unix. Normalmente, grep toma una expresión regular de la línea de comandos, lee la entrada estándar o una lista de archivos, e imprime las líneas que contengan coincidencias para la expresión regular.

Su nombre deriva de un comando en el editor de texto ed que tiene la siguiente forma: g/re/p

Existen otros retroacrónimos (incorrectos) para el nombre, entre ellos: General Regular Expression Parser (analizador general de expresiones regulares), General Regular Expression Print (imprimir expresión regular general), y Global Regular

Expression Print (imprimir expresión regular global), éste último no tan lejano de la realidad.

Grep generalmente ejecuta alguna variante del algoritmo Boyer-Moore (para búsqueda de strings), utilizando expresiones regulares para definir la consulta. Puede manejar archivos, directorios (y subdirectorios), o la entrada estándar (stdin).

El programa es configurable por medio de opciones de invocación, pudiendo (por ejemplo) mostrar las líneas con aciertos, desaciertos, el contexto de la coincidencia, etc.

10. Dd

DD es un comando Unix muy popular cuya principal función es la conversión de datos raw y la copia de datos byte a byte. DD es un comando que "convierte y copia sistemas de ficheros" basándose en el protocolo de la versión 7 de UNIX desarrollado por IBM.

DD es utilizado para copiar un numero especifico de bytes o bloques de memoria (utilizando para ello la copia al vuelo), para copiar discos duros y para leer o recuperar archives dañados. Su sintaxis en nuestro caso es la siguiente (aunque posee más opciones y funcionalidades):

if=*FILE*
Input file : el comando lee desde file el archivo a copiar (lugar origen).

of=*FILE*
Output file : el commando escribe en file el archivo a copiar (lugar destino)

11. Mkdir

Mkdir es una orden de los sistemas operativos UNIX, DOS, OS/2 y Microsoft Windows usada para crear un nuevo directorio o carpeta del sistema de archivos.

Mkdir tiene su origen en las palabras make directory (crear directorio en ingles). De las múltiples opciones que ofrece este comando, es de particular interés la opción –m:

-m, --mode Establece el tipo de permisos (al igual que chmod) que poseerá la carpeta.

12. Cd

cd (del inglés change directory), es un comando que nos permite cambiar el directorio de trabajo actual en los sistemas operativos basados en UNIX (aunque también funciona en Windows). También es posible su uso en shell scripts y archivos batch (que es el uso que se le ha dado en la realización de este proyecto).

13. Chmod

Chmod ("change mode", cambiar modo en inglés) es una llamada al sistema y su comando asociado en el sistema operativo UNIX (estandarizados en POSIX y otros estándares) que permite cambiar los permisos de acceso de un archivo o directorio.

Existen tres permisos independientes, llamados permisos básicos, que pueden ser permitidos (símbolo + precediéndolos) o denegados (símbolo - precediéndolos) a un archivo y/o directorio

- r - lectura
- w - escritura
- x - ejecución

14. Apt-get install

Advanced Packaging Tool (Herramienta Avanzada de Empaquetado), abreviado APT, es un sistema de gestión de paquetes creado por el proyecto Debian. APT simplifica en gran medida la instalación y eliminación de programas en los sistemas GNU/Linux.

No existe un programa apt en sí mismo, sino que APT es una biblioteca de funciones C++ que se emplea por varios programas de línea de comandos para distribuir paquetes.

Anexo 2:

Particiones, características y tipos

1. Introducción

Una partición de disco, en informática, es el nombre genérico que recibe cada división presente en una sola unidad física de almacenamiento de datos. Toda partición tiene su propio sistema de archivos (formato). Generalmente, casi cualquier sistema operativo interpreta, utiliza y manipula cada partición como un disco físico independiente, a pesar de que dichas particiones estén en un solo disco duro.

2. Información general

Las particiones, por decirlo de algún modo, son los elementos en que está dividido un disco duro. Así, cuando decimos que un disco duro tiene tres particiones, significará que el disco duro está dividido en tres unidades (no necesariamente iguales). Dicho de otra manera, está dividido en tres unidades lógicas (las particiones) dentro de una unidad física (el disco duro).

Todo disco duro que deba ser utilizado con cualquier sistema operativo deberá tener como mínimo una partición. Si sólo tenemos una significará que el disco duro no está dividido, y la partición ocupa todo el disco duro quitando el espacio no particionado. Nosotros podemos elegir el tamaño de cada partición en función de lo que irá en cada una de ellas y de la capacidad de tamaño del disco duro. En caso de que sobre espacio, será denominado espacio no particionado. El espacio no particionado de un disco duro no es accesible al no formar parte de ninguna partición.

Supongamos que en nuestro ordenador tenemos un sistema operativo instalado y que el ordenador lo utiliza más de una persona, podríamos crear varias particiones lógicas a las cuales se pueda acceder desde el sistema operativo como si fueran discos duros diferentes, Guardando de esta manera cada uno sus datos y su información en una partición distinta.

Los sistemas operativos se deben instalar en particiones primarias. Si tenemos más de una partición primaria con un sistema operativo en cada una, debemos establecer como activa aquella que tenga el sistema operativo con el que queramos empezar a utilizar el ordenador.

Si quisiéramos tener dos sistemas operativos o más en un ordenador con un único disco duro, deberemos particionar el disco duro de manera que obtengamos una partición para cada sistema operativo. Eso no significa que no se puedan instalar varios

sistemas operativos en la misma partición, pero esto último no se recomienda ya que puede dar lugar a errores.

Además, a cada partición le deberemos asignar un sistema de archivos (diferente en cada sistema operativo) especifico. Un sistema de archivos es el método para nombrar, almacenar y organizar archivos en el equipo.

Existen diferentes tipos de particiones (primaria, lógica, extendida) y sistemas de archivos (FAT, FAT32, NTFS, EXT2, EXT3). Procederemos a explicarlas en los dos apartados siguientes.

3. Tipos de particiones

El formato o sistema de archivos de las particiones (p. ej. NTFS) no debe ser confundido con el tipo de partición (p. ej. partición primaria), ya que en realidad no tienen directamente mucho que ver. Independientemente del sistema de archivos de una partición (FAT, ext3, NTFS, etc.), existen 3 tipos diferentes de particiones:

- **Partición primaria**: Son las divisiones crudas o primarias del disco, solo puede haber 4 de éstas o 3 primarias y una extendida. Depende de una tabla de particiones. Un disco físico completamente formateado consiste, en realidad, de una partición primaria que ocupa todo el espacio del disco y posee un sistema de archivos. A este tipo de particiones, prácticamente cualquier sistema operativo puede detectarlas y asignarles una unidad, siempre y cuando el sistema operativo reconozca su formato (sistema de archivos).
- **Partición extendida**: Es otro tipo de partición que actúa como una partición primaria; sirve para contener infinidad de unidades lógicas en su interior. Fue ideada para romper la limitación de 4 particiones primarias en un solo disco físico. Solo puede existir una partición de este tipo por disco, y solo sirve para contener particiones lógicas. Por lo tanto, es el único tipo de partición que no soporta un sistema de archivos directamente.
- **Partición lógica**: Ocupa una porción de la partición extendida o la totalidad de la misma, la cual se ha formateado con un tipo específico de sistema de archivos (FAT32, NTFS, ext2,...) y se le ha asignado una unidad, así el sistema operativo reconoce las particiones lógicas o su sistema de archivos. Puede haber un máximo de 23 particiones lógicas en una partición extendida.

4. Tipos de Sistemas de archivos

Dado que el objetivo final de este anexo es introducir o guiar al usuario inexperto y resolver posibles dudas que surjan durante la lectura de esta memoria, solo explicaremos los dos sistemas de archivos utilizados en este proyecto. El sistema NTFS que utiliza el SO Windows, y el sistema de archivos ext4 propio de Linux.

EXT4 (fourth extended filesystem o «cuarto sistema de archivos extendido») es un sistema de archivos con registro por diario (en inglés Journaling), anunciado el 10 de octubre de 2006 por Andrew Morton, como una mejora compatible de ext3.

Las principales mejoras son:

- Soporte de volúmenes de hasta 1024 PiB.
- Soporte añadido de extent.
- Menor uso del CPU.
- Mejoras en la velocidad de lectura y escritura.

NTFS (NT File System) es un Sistema de archivos de Windows NT incluido en las versiones de Windows 2000, Windows XP, Windows Server 2003, Windows Server 2008, Windows Vista y Windows 7. Está basado en el sistema de archivos HPFS de IBM/Microsoft usado en el sistema operativo OS/2, y también tiene ciertas influencias del formato de archivos HFS diseñado por Apple.

NTFS permite definir el tamaño del clúster, a partir de 512 bytes (tamaño mínimo de un sector) de forma independiente al tamaño de la partición.

Es un sistema adecuado para las particiones de gran tamaño requeridas en estaciones de trabajo de alto rendimiento y servidores. Puede manejar volúmenes de, teóricamente, hasta 2^{64}–1 clústeres. En la práctica, el máximo volumen NTFS soportado es de 2^{32}–1 clústeres (aproximadamente 16 Terabytes usando clústeres de 4KB).

5. El área de intercambio (Swap)

La memoria swap es un espacio reservado en el disco duro para poder usarse como una extensión de memoria virtual del sistema. Es una técnica utilizada desde hace mucho tiempo para hacer creer a los programas que existe más memoria RAM de la que en realidad existe.

El propio sistema operativo es el encargado de pasar datos a la memoria swap cuando necesita más espacio libre en la RAM y viceversa.

En Linux, la memoria total disponible por el sistema está formada por la cantidad de memoria RAM instalada más la swap disponible. El acceso a la swap (disco duro) es más lento que el acceso a la memoria RAM, por lo que si el ordenador posee una carga de trabajo grande y hace un uso intensivo de la swap, la velocidad del sistema disminuirá. Un uso muy intensivo y continuado de la swap es un indicativo de que necesitamos más memoria en nuestro sistema para que funcione acorde con el uso que le estamos dando.

En Linux generalmente se usa como mínimo una partición dedicada a swap (aunque también se puede tener un fichero swap). El tamaño recomendado de la misma es el mismo que el de la memoria RAM que posee el sistema.

Bibliografía

Guía Para Administradores de Sistemas GNU/Linux , *de Lars Wirzenius, Joanna Oja, Stephen Stafford, Alex Weeks. Traducción de la versión 0.8 del libro The Linux System Administrator's Guide realizada por Rafael Ignacio Zurita.*

Guía de Administración de Redes, *de Olaf Kirch. Traducción de la versión 1.0 del libro Linux Network Administration Guide.*

Guía Linux de Programación, *de Sven Goldt. Traducción de la versión 0.4 del libro Linux Programmer's Guide.*

Guía del Usuario de Linux, *de Larry Greenfield . Traducción del libro Linux User's Guide.*

Advanced Bash-Scripting Guide, *de Mendel Cooper.*

Bash Guide for Beginners, *de Machtelt Garrels.*

An A-Z Index of the Bash command line for Linux, *documentation HTML de todos los commandos de bash.* http://ss64.com/bash/.

GNU/Linux Command-Line Tools Summary, *de Gareth Anderson.*

Introducción a Secure Shell, de Javier Smaldone. Puede ser consultado en formato .pdf en http://es.tldp.org/Tutoriales/doc-ssh-intro/introduccion_ssh-0.2.pdf.

Open ssh man pages, documentación HTML de todo lo referente a SSH. http://www.openssh.com/manual.html.

Administracion de una red básica en entorno GNU/LINUX, *memoria de proyecto fin de carrera realizada por Ekaitz Hernandez Troyas, estudiante de la upna.*

La gran mayoría de bibliografía utilizada durante la realización de este proyecto no posee copyright y es posible su lectura o descarga gratuita en paginas como “The Linux documentation Project” (http://tldp.org/guides.html).

También ha de destacarse en este apartado, a pesar de no ser libros en si mismos, la información hallada en los siguientes foros de internet:

- http://forums.debian.net/
- http://www.unix.com/
- http://www.esdebian.org/
- http://ubuntuforums.org/