POLITÉCNICA

Script maestro para la configuración de un cluster Linux

CABALLERO PASTOR SANTIAGO FEDERICO
DEL RINCON GARCIA MARIO
RODRIGUEZ GUTIERREZ NACHO
Grupo 19

1.- Introducción

Este proyecto consiste en un conjunto de scripts que facilitan la labor de un administrador de clusters formados por varias máquinas con sistema Linux.

El funcionamiento consiste en un script general que recibe por parámetro un fichero de configuración en el que vienen las distintas ip's,servicios y ficheros de configuración específicos para realizar las tareas. Hay un script auxiliar para cada servicio que se ha implementado.

En esta memoria explicamos al detalle el funcionamiento de estos.

Para que las soluciones propuestas funcionen, es necesario que las máquinas cumplan unos requisitos iniciales:

- Todas las máquinas deben tener disponible el servidor **SSH**.
- Las máquinas deben tener la misma configuración
- Las máquinas deben tener distintas direcciones MAC.
- Los scripts deben estar en el mismo directorio.
- Debe ser ejecutado en el directorio donde se encuentran los scripts.

2.- Script General: configurar_cluster.sh

El script general será el script "maestro" que llamará a los demás scripts. Para llamar a este script, hay que usar el comando (importante, debe ser en superusuario):

./configurar_cluster.sh fichero_configuracion

Este script recibe como parámetro un fichero de configuración con un formato como el de la captura siguiente:

```
#Esto es un comentario

192.168.0.1 raid raid.conf
192.168.0.1 mount mount_raid.conf
192.168.0.1 nfs_server nfs_server.conf
```

- IP de la máquina donde queremos realizar el servicio
- Servicio que queremos realizar en esa máquina
- Fichero de configuración adicional para realizar el servicio

El script primero comprobará que el fichero de configuración haya sido pasado por parámetro, además se asegurará de que existe y tiene datos. Ignorará los saltos de línea y los comentarios (líneas que empiezan por el carácter "#"). Leyendo únicamente las líneas

que contengan los datos que se muestran en la captura y comprobará que se sigue la sintaxis que se muestra en la captura (IP, SERVICIO, FICHERO).

En caso de que el servicio no se encuentre dentro de los que ofrece la práctica dará un error, parando la ejecución y devolviendo un valor 99.

Por tanto, el fichero va leyendo línea a línea el fichero de configuración. Cuando encuentra una línea con Ip, servicio y fichero adicional, llama a la función auxiliar creada dentro de este script general llamada "**iniciar**" pasándole como parámetros tanto la IP, como el nombre del servicio y el nombre del fichero de configuración adicional. Decidimos crear esta función porque a medida que fuimos avanzando en el proyecto nos dimos cuenta de que por cada servicio que programábamos se duplicaba código. Además, esto daba lugar a un código mucho más estructurado y legible.

La función **iniciar** copia este fichero adicional en la máquina en la que se quiere realizar el fichero. En caso de que no se haya copiado correctamente por cualquier error, terminará la ejecución y devolverá un valor distinto de 0.

Tras copiar el fichero adicional en la máquina correspondiente, ejecuta a través de ssh el script auxiliar del servicio correspondiente en la máquina remota pasándole como parámetro el fichero que hemos copiado anteriormente.

El script del servicio hará las comprobaciones oportunas en cada caso particular, además de ejecutar todo lo necesario para realizar las configuraciones queridas.

Cuando termina el script auxiliar, borra el fichero de configuración auxiliar que había copiado en la máquina remota, comprueba el valor que ha devuelto el script del servicio y lo retorna al bucle principal del que fue llamada.

En caso de que sea distinto de 0 lo que ha devuelto la función iniciar, termina la ejecución. Si es 0, sigue leyendo el fichero de configuración.

En cuanto a la sintaxis de todos los ficheros de configuración, es importante resaltar que se deje siempre una **línea vacía al final**, ya que si no la última línea no sería leída. Además, también es obligatorio que los ficheros de configuración estén en **formato Unix** para evitar posibles errores por el carácter CR que tienen los ficheros en formato Windows.

Ejemplo de formato incorrecto:

```
#Esto es un comentario CRLF
CRLF
192.168.0.1 raid raid.conf CRLF
192.168.0.1 mount mount_raid.conf CRLF
192.168.0.1 nfs_server nfs_server.conf
```

La última línea no sería leída, además de que el carácter CR daría probablemente problemas.

Ejemplo de formato correcto:

```
#Esto es un comentario LF
LF
192.168.0.1 raid raid.conf LF
192.168.0.1 mount mount_raid.conf LF
192.168.0.1 nfs_server nfs_server.conf LF
```

En este ejemplo sí se leería la última línea y está en formato Unix.

Los servicios que realiza son los siguientes:

- Mount
- Lvm
- Raid
- NIS
- NFS
- Backup

3.- Script auxiliares.

A continuación, vamos a explicar lo que hace cada script creado para cada servicio. Como norma general, en ellos se comprueba que el fichero de configuración adicional para el servicio tenga una sintaxis correcta.

3.-1 Servicio MOUNT: servicio_mount.sh

En este servicio primero se comprueba que el fichero de configuración sea correcto, teniendo dos líneas, una para el dispositivo donde se va a realizar el mount y otra para el punto de montaje.

Tras esta comprobación, mira si existe el directorio pasado como punto de montaje. En caso de que este no exista lo crea y en caso de que existe comprueba si está vacío, si no lo está devuelve un error y el aviso de que se necesita un directorio vacío.

Después se encarga de ver que el dispositivo exista y realiza el mount del directorio en él. Si todo ha ido bien, como se quiere que este montaje exista cada vez que se enciende la máquina, obtiene el UID del montaje y añade la línea correspondiente al fichero /etc/fstab devolviendo un 0 si todo ha salido correctamente.

Problemas encontrados durante la realización de este script: en el momento de hacer el mount, lo ejecutaba directamente con el parámetro -t y diciendo que era de tipo ext4, cuando debía haberlo hecho con la opción auto.

3.-2 Servicio RAID: servicio_raid.sh

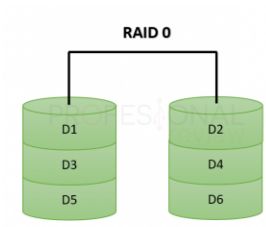
Para este servicio se recibe un fichero de configuración donde nos importan tres líneas: el nombre del dispositivo raid que vamos a montar, el nivel de raid, y los dispositivos usados para ese raid. Por tanto lo que hace este script es ignorar todas las líneas vacías y comentarios y comprobar que efectivamente existen esas 3 líneas de la manera correcta.

Tras terminar la comprobación del fichero de configuración, comprueba que todos los dispositivos pasados en la tercera línea existen y que no contienen ningún sistema de ficheros en ellos.

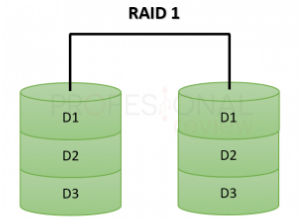
Una vez que se comprueba que existen, se asegura de que el número de dispositivos que se han pasado por parámetro se ajustan al nivel de Raid que se ha pedido realizar.

Hay varios niveles de Raid y estos son los que se han implementado en la práctica:

RAID 0

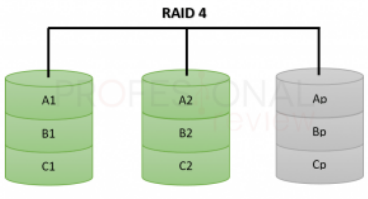


RAID 1

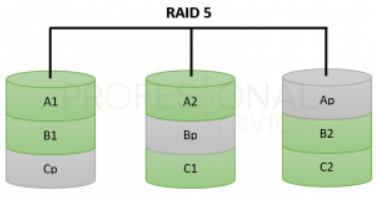


Para estos dos niveles de Raid (0 y 1), se necesitan mínimo dos dispositivos. El servicio comprueba que en la lista recibida haya al menos dos y en caso contrario, manda error.

RAID 4

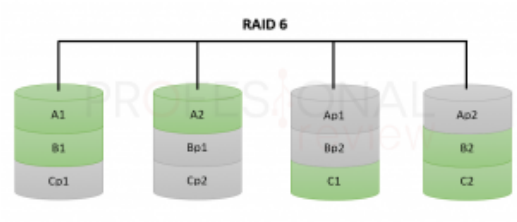


RAID 5

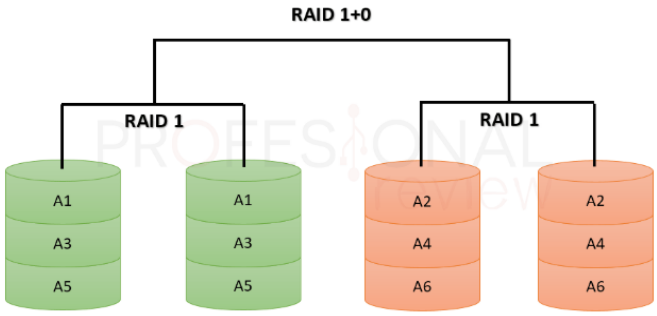


Para el nivel de Raid 4 y nivel de Raid 5 se necesitan al menos tres dispositivos. Por lo que comprobará que haya al menos 3 en la lista.

RAID 6



RAID 1+0



Y para terminar, los niveles de Raid 6 y 1+0 (en la práctica conocido como nivel de Raid 10), que necesitan ambos 4 dispositivos mínimo pasados en la lista.

En cualquiera de los niveles en los que se pase menos dispositivos de los que se necesita, terminará la ejecución del script y devolverá por pantalla un mensaje especificando el nivel que se ha pedido y el número mínimo de dispositivos que se necesitan para éste, devolviendo un error con código 60.

En caso de que se pase un nivel de Raid que no está entre los mencionados, dará un error con código 30 y saldrá por pantalla un mensaje avisando de ello.

Tras esto, instala el mandato mdadm en la máquina donde se realiza el Raid (en caso de que no lo tenga ya instalado) y crea el Raid. Si se ha creado con éxito, se añade al fichero /etc/mdadm/mdadm.conf y si se añade correctamente, termina la ejecución del servicio devolviendo un código 0.

Problemas encontrados durante la realización de este script: tuvimos dificultades a la hora de saber qué niveles de Raid había que implementar y cuántos dispositivos se necesitaban para cada uno de ellos.

3.-3 Servicio LVM: servicio_lvm.sh

Para este servicio se recibe un fichero de configuración con mínimo 3 líneas. El servicio comprueba ese mínimo de líneas y tras ello realiza un bucle leyendo línea a línea hasta el final. Las dos primeras líneas consisten en el nombre del grupo de volúmenes y la lista de dispositivos que se usará para el volumen lógico.

Una vez leídas todas las líneas, se instala en la máquina el mandato lvm2.

Si todo ha ido bien (el fichero de configuración y la instalación del mandato) comprueba que todos los dispositivos de la lista existen y están disponibles. Si lo están, se inicializa un volumen físico con ellos con **pvcreate** y se crea el grupo con **vgcreate.** Si todo ha funcionado correctamente, comprueba que el tamaño de ese volumen físico creado es suficiente para los volúmenes lógicos que se quieren crear y si es así se crean todos los volúmenes lógicos recorriendo un array donde se almacenaron. Para crear ese volumen lógico se usa el mandato **lvcreate.**

Problemas encontrados durante la realización de este script: no tuvimos en cuenta que pudieran pasarse distintos prefijos de unidad de medida, por lo que no pasábamos a la misma unidad y nuestro cálculo del volumen lógico total que se pedía no era correcto. Además, hay algunas pruebas donde el valor neto del volumen físico es insuficiente para el volumen lógico que se requiere mientras que el valor bruto sí que lo es y no lo detectamos. Tampoco detectamos que se pasaran mal el tamaño del volumen lógico que se quería crear.

3.-4 Servicio servidor NFS: servicio_nfs_server.sh

En este servicio, primeramente, se comprueba que el fichero de configuración no esté vacío. Si está vacío, el servicio acabará con un valor de retorno distinto de 0. Si no está vacío, se comprueba que existan los directorios especificados en las líneas del fichero de configuración. Para ello, se realiza un bucle. En el bucle, en primer lugar, se averigüa si las líneas tienen un solo valor, si no es así, el servicio acabará con un valor de retorno distinto de 0. En segundo lugar, una vez verificado que cada línea tiene un solo valor, se muestra que existen los directorios. Si existen todos los directorios, se introducen en un array y se procede con la instalación de nfs con el mandato “apt-get -y install nfs-kernel-server”. A continuación, se sustituye el valor del dominio por el de la red modificando el fichero “/etc/idmapd.conf” y se configuran los recursos compartidos editando el archivo “/etc/exports” mediante un bucle for, introduciendo los directorios pasados por el fichero de configuración al final del fichero, verificando que no haya ningún error en los mandatos de sustitución, “sed”. Después, hacemos efectivos los cambios producidos al fichero

“/etc/exports” mediante el comando “exportfs -ra”, verificando que no haya ningún error en los mandato.

3.-5 Servicio servidor NFS: servicio_nfs_client.sh

En este servicio, primeramente, se comprueba que el fichero de configuración no esté vacío. Si está vacío, el servicio acabará con un valor de retorno distinto de 0. Si no está vacío, se comprueba que existan los sistemas de ficheros que deben montarse especificados en las líneas del fichero de configuración. Para ello, se realiza un bucle. En el bucle, en primer lugar, se averigüa si las líneas tienen solo tres valores, si no es así, el servicio acabará con un valor de retorno distinto de 0. En segundo lugar, una vez verificado que cada línea tiene tres valores, se muestra que el segundo valor sea un directorio vacío y si no existe se crea acabando el bucle guardando las líneas en un array. Una vez completado, se realiza el montaje y añadimos a “/etc/fstab” para que se mantenga. Con este fin, se ejecuta un bucle for en el que por cada línea del fichero de configuración, se hace un montaje, el primer valor es la IP del servidor, el segundo valor el directorio a exportar y el tercer valor el punto de montaje. Todo ello verificando que no haya ningún error en el mandato mount -t nfs. Por último añadimos al final del fichero “/etc/fstab” la IP del servidor, el directorio a exportar y el punto de montaje de todos los montajes, finalizando así el bucle y el servicio.

3.-6 Servicio servidor NIS: servicio_nis_server.sh

En este servicio, primeramente, se comprueba que el fichero de configuración contenga una línea con un solo valor que sería el nombre del dominio NIS. Una vez demostrado esto, se procede a instalar el paquete NIS con “apt-get install nis”, ayudándonos de “mount --bind /bin/true /usr/sbin/invoke-rc.d” para que el proceso de instalación sea más rápido. Cuando acaba la descarga, se comprueba que no haya fallado y desmontamos, “umount /usr/sbin/invoke-rc.d”. A continuación, modificamos el contenido de los ficheros “/etc/defaultdomain”, “/etc/default/nis”, “/etc/yp.conf”, y “/var/yp/Makefile” verificando que no haya ningún error en los mandatos de sustitución, “sed”. En el primer fichero, ponemos el dominio pasado por el fichero de configuración. En el segundo fichero, cambiamos “NISSERVER=master”, para establecer a la máquina como maestra. En el tercer fichero, añadimos el dominio del servidor al final del fichero escribiendo “domain $DOMINIO_NIS server `hostname` ”. En el último fichero, para poder hacer un login en el cliente usando los datos del repositorio, permitimos que la información de /etc/shadow y /etc/gshadow se mezcle, respectivamente, con la de /etc/passwd y /etc/group. Por ello ponemos a verdadero “MERGE_PASSWD” y “MERGE_GROUP”. Por último, reiniciamos el servicio NIS con “systemctl restart nis &” y comprobamos que no haya ocurrido ningún error. Esperamos un segundo e introducimos /dev/null en “/usr/lib/yp/ypinit -m” para no

permitir esclavos, y esperamos a que se acabe de reiniciar el servicio NIS. Con ello, se tendría el servidor NIS.

Problemas encontrados durante la realización de este script: al principio, este servicio fue confuso para nosotros ya que no conseguimos entenderlo, pero tras la ayuda del profesorado y la documentación proporcionada, conseguimos comprender el servicio y entender cómo funcionaba. Un error, más puntual, que ocurrió tanto en la parte cliente como en la parte servidor fue el menú que saltaba cuando instalabas el servicio nis para introducir el dominio nis. Este menú no supimos que debíamos quitarlo hasta que se nos lo indico y tras una profunda búsqueda de información, conseguimos solucionarlo.

3.-7 Servicio cliente NIS: servicio_nis_client.sh

En este servicio, primeramente, se comprueba que el fichero de configuración contenga dos líneas cada una de ellas con un solo valor, y que la segunda tenga formato de ip. Una vez demostrado esto, se procede a instalar el paquete NIS con “apt-get install nis”, ayudándonos de “mount --bind /bin/true /usr/sbin/invoke-rc.d” para que el proceso de instalación sea más rápido. Cuando acaba la descarga, se comprueba que no haya fallado y desmontamos, “umount /usr/sbin/invoke-rc.d”. A continuación, modificamos el contenido de los ficheros “/etc/defaultdomain”, “/etc/yp.conf”, y “/etc/nsswitch.conf” verificando que no haya ningún error en los mandatos de sustitución, “sed”. En el primer fichero, ponemos el dominio pasado por el fichero de configuración en la primera línea. En el segundo fichero, añadimos el dominio del servidor al final del fichero escribiendo “domain $DOMINIO_NIS server $IP ”. En el último fichero, permitimos el login NIS, modificando “passwd: files systemd” a ”passwd: compat systemd nis”. Por último, reiniciamos el servicio NIS con “systemctl restart nis” y comprobamos que no haya ocurrido ningún error.

3.-8 Servicio servidor BackUp: servicio_backup_server.sh

En este servicio, primeramente, se comprueba que el fichero de configuración contenga una línea con un único valor que sería el nombre del directorio donde se realizará el BackUp. Una vez demostrado esto, se comprueba que el directorio existe. Si no existe se crea, y si existe se comprueba que está vacío. Si no está vacío, se transmitirá el error.

Problemas encontrados durante la realización de este script: la comprobación de que el directorio local existiese y estuviese vacío, no se hizo adecuadamente.

3.-9 Servicio cliente BackUp: servicio_backup_client.sh

En este servicio, primeramente, se comprueba que el fichero de configuración contiene cuatro líneas con un único valor cada una. Una vez demostrado esto, primero se demuestra que el primer valor es un directorio local. Segundo, se comprueba que el segundo valor sea una ip con el formato correcto. Tercero, que el tercer valor sea un directorio remoto. Cuarto, se comprueba que el cuarto valor sea un número entre 1 y 23. Si todas estas comprobaciones son exitosas, se procede a llamar al mandato rsync para poder realizar el BackUp dentro de contrab: “sed -i -e "\$a 0 */$HORAS * * * root rsync --recursive $DIRECTORIO_LOCAL root@$DIRECCION:$DIRECTORIO_REMOTO" /etc/crontab”.

Hemos elegido este método por una parte porque era el recomendado por los profesores de la asignatura y por otra parte, y que se trata de un servicio con enfoque cliente-servidor ideal para la realización del BackUp en el proyecto.

Problemas encontrados durante la realización de este script: no implementamos correctamente el demonio cron usando el archivo /etc/crontab.