Proyecto ASI 2015

Script Maestro para configuración de un clúster en Linux

UPM

4 de julio de 2016

Autores: Jorge Amorós (120200) y Antonio Carrizosa (150505)

Tabla de contenido

1. Objetivo del proyecto

A lo largo de este documento se va a documentar la realización de un script encargado de ejecutar un conjunto de servicios en un clúster de máquinas sin que la participación del usuario sea requerida. Este script recibirá un fichero indicando que servicios debe configurar y en que máquina debe hacerlo.

Para esto, se ha optado por una estructura modular, es decir, cada servicio está desarrollado en un archivo diferente. Esto facilitará su modularidad y su legibilidad. Para su ejecución en una máquina del clúster, este es copiado en un directorio temporal de dicha máquina, ejecutado y posteriormente eliminado. Cada script de servicio está acompañado de un fichero con los parámetros necesarios para su correcta configuración correcta de dicho servicio.

Los nombres de estos scripts son los siguientes, y tienen que estar presentes a la hora de ejecutar el script principal configurar_cluster.sh.

Script_mount.sh
Script_raid.sh
Script_lvm.sh
Script_nis_server.sh
Script_nis_client.sh
Script_nfs_server.sh
Script_nfs_client.sh
Script_backup_server.sh
Script_backup_client.sh

Las instrucciones para la ejecución de script serían las siguientes:

1. Crear el correspondiente fichero de configuración como se indica en las instrucciones del proyecto. Máquina-Servicio-FicheroAdicionaldeConfiguracion

2. Rellenar el fichero adicional en función del servicio que vamos a invocar tal y como se indica en la guía de configuración.

3. Ejecutamos configurar_cluster.sh con el fichero de configuración como argumento.

En principio debería de ejecutarse automáticamente y sin errores siempre y cuando la configuración ssh no requiera contraseña y este configurar para hacer ssh como root.

IMPORTANTE: Los scripts fueron programados en Windows usando sublime text, lo que puede dar lugar a conflictos cuando los exportas a unix para su ejecución. Han sido probados todos repetidamente, pero es posible, debido a las diversas copias de seguridad, que se me haya colado uno sin haberlo trasformado del formato que utiliza Windows a unix y no por lo tanto a la hora de ejecutarse no reconozca el intérprete. Sería raro como he mencionado, pero si esto ocurriera habría que copiarlo y pegarlo a un nuevo archivo en el UNIX. Lamento las molestias, aunque confío en que esto no pase.

Otra solución más sencilla seria ejecutar

```
sed -i -e 's/\r$//' scriptname.sh
```

2.Servicios

En este apartado vamos a realizar una explicación detallada del funcionamiento de los servicios. El comienzo de todos ellos es muy similar, comprobando que tiene el número de líneas adecuado y comprobando también algún patrón particular como direcciones ip.

Configurar_cluster.sh

Este script es el principal, y el encargado de llamar a los demás scripts de configuración. Para esto recibe un fichero de configuración y se encarga de hacer las comprobaciones necesarias. Primeramente, eliminará todos los comentarios y las líneas en blanco, dejando solamente las líneas representativas. Para cada línea comprobara que contiene 3 parámetros, comprobando siempre que el primero es una dirección ip correcta y que el segundo es un servicio correcto. Entendemos que el tercer parámetro se puede llamar de cualquier manera así que no hay comprobación sobre este. Si hay algún error, el script se detiene indicando la línea donde se ha producido el error. Posteriormente, se encarga de ir llamando a los servicios, conectándose como root vía ssh a la maquina destino, copiando el script en una carpeta temporal y dándole permisos, así como su fichero de configuración y ejecutándolo. Borrando todo al final del proceso.

Problemas encontrados

Probablemente es más complejo de todos y el más difícil debido a que hacía tiempo que no programábamos en bash scripting. Conseguir el filtrado de todas las líneas fue complejo, al final se hizo mediante el uso de sed, pero hubo que trabajar bastante para encontrar la solución. El bucle para recorrer cada línea y cada palabra de esta también fue complejo de realizar. La parte de conexión ssh fue más

sencilla debido a que hay bastantes ejemplos en la red y es algo más genérico que realizar.

Pero el fallo de última hora y fue que en el último bucle donde exportarnos remotamente los scripts y sus ficheros a las maquinas destino, solamente hacia una iteración del bucle. Esto es así porque read lee de la salida estándar y ssh también, comiéndose el resto de las iteraciones y finalizando.La solución fue encontrada en un post que dejamos a continuación al resultarnos algo curioso.

"http://unix.stackexchange.com/questions/107800/using-while-loop-to-ssh-to-multiple-servers"

Script_mount.sh

Este script se encarga de montar el dispositivo indicado en el directorio indicado. Hará las comprobaciones pertinentes al comienzo, entre ellas: número de líneas correcta, dispositivo existente, directorio de montado vacío o directorio existente. Posteriormente y en función de lo comprobado el script o bien se aborta o bien configura el montaje. Si configura correctamente el montaje escribirá en fstab la orden para el montaje automático al comienzo de cada sesión, comprobando previamente que no se encontraba dicha orden en fstab.

Problemas encontrados

El primer problema fue sin duda la comprobación de que un dispositivo existe porque si bien te pueden pasar un dispositivo tal que /dev/sda y utilizar el cut para tomar el ultimo valor y comprobar si existe en proc/partitions, como hacer si te pasan un volumen lógico para coger la última parte. Para esto se usó un rev al principio un cut normal y otro rev al final. Para comprobar si había ya algo montado, no se sabía muy bien cómo hacerlo hasta que encontramos el mandado mountpoint que nos proporciona dicha utilidad. El patrón para filtrar el fstab también fue bastante complejo de realizar.

También hubo que buscar la manera de comprobar el número de líneas de un fichero para ver si es correcto. Damos por hecho que no hay comentarios en estos ficheros de configuración. Se usó el wc -l <$1, hincapié especial en ese menor para que solo se obtenga el número de líneas y no también el nombre del fichero.

Script_raid.sh

Este script se encarga de realizar un raid de los dispositivos pasados como parámetro. Hará las comprobaciones pertinentes al comienzo, entre ellas: número de líneas correctas o el tipo de raid correcto. Posteriormente y en función de lo comprobado abortará el script o se continua y se configura el raid.

Problemas encontrados

Al tener que instalar la herramienta mdadm para generación de raid, fue muy complicado conseguir que esa instalación no pidiese cierta gestión de usuario. Tras mucho buscar se consiguió exportando la variable DEBIAN_FRONTEND=noninteractive. No estamos seguros de que hace exactamente, pero funciona. Esta fue la única complicación real para este script.

Script_lvm.sh

Este script se encarga de la creación de los volúmenes lógicos indicados. Hará las comprobaciones pertinentes al comienzo, entre ellas: número de líneas correctas. Posteriormente y en función de lo comprobado abortará el script o configurará los volúmenes lógicos.

Problemas encontrados

El problema principal fue determinar la estructura del script puesto que el número de argumentos recibidos es variable. Una vez determinado esto el resto fue trivial.

Script_nis_server.sh

Este script se encarga de la creación de un servidor nis con el nombre pasado por parámetro encargado del envío de datos de comunicación. Hará las comprobaciones pertinentes al comienzo, entre ellas: número de líneas correctas. Posteriormente se encarga de configurar el servidor modificando los archivos /etc/defaultdomain, etc/default/nis y /etc/yp.conf. Finalmente se reinicia el servicio nis.

Problemas encontrados

El principal problema encontrado fue el uso de sed para modificar un archivo. Tuvimos que investigar bastante para ver cómo hacer los cambios permanentes y como se usan los patrones. También el uso de ypinit -m, no sabíamos cómo ejecutarlo para que se guarden los cambios y no se quede en bucle que es como nos sucedía. Para ello hicimos un pipe con un EOF o Control+D.

Script_nis_client.sh

Este script se encarga de la configuración de un cliente para el uso de un servidor nis. Hará las comprobaciones pertinentes al comienzo, entre ellas: número de líneas correctas, dirección ip correcta…En función de lo comprobado, abortará el script o configurará el cliente. Para esto modificara los ficheros /etc/yp.conf , etc/default/nis y etc/nsswitch.conf.

Problemas encontrados

Los problemas encontrados fueron muy similares a los encontrados en script_server.sh

Script_nfs_server.sh

Este script se encarga del montaje de los directorios remotos pasados como parámetro. Hará las comprobaciones pertinentes al comienzo, entre ellas: número de líneas correctas. En función de lo comprobado, abortará el script o montará todos los directorios remotos sobre los directorios indicados. Si algún directorio local no existe se crea.

Problemas encontrados

Ningún problema de intereses destacado o no mencionado anteriormente.

Script_nfs_client.sh

Este script se encarga de la configuración del directorio etc/exports donde guardaremos los directorios a exportar. máquina. Hará las comprobaciones pertinentes al comienzo, entre ellas: número de líneas correctas. En función de lo comprobado, abortará el script o configurará todos los directorios a exportar pasados como parámetro en el fichero.

Problemas encontrados

Ningún problema de intereses destacado o no mencionado anteriormente.

Script_backup_server.sh

Este script se encarga de la configuración de directorio donde se guardarán las copias para el backup desde otra máquina. Hará las comprobaciones pertinentes al

comienzo, entre ellas: número de líneas correctas, directorio destino existe y vacío…En función de lo comprobado, abortará el script o saldrá exitosamente puesto que el directorio es correcto.

Problemas encontrados

Ninguno realmente, al buscar información para comprobar si el directorio estaba vacío, también encontramos la opción para buscar también archivos ocultos. Está bien indicarlo.

Script_backup_client.sh

Este script se encarga de la configuración de una tarea periódica de backup de un directorio propio, en la máquina, directorio y con una frecuencia indicada en el fichero de configuración. Hará las comprobaciones pertinentes al comienzo, entre ellas: número de líneas correctas, IP destino correcta, directorio destino existente…En función de lo comprobado, abortará el script o escribirá en etc/crontab el mandato indicado para establecer este backup periódico. No se especifica así que no hemos comprobado si la orden ya estaba en crontab. Damos por hecho que no van a repetirse órdenes.

Problemas encontrados

Ninguno realmente, al buscar información para comprobar si el directorio estaba vacío, también encontramos la opción para buscar también archivos ocultos. Está bien indicarlo.

3.Conclusiones

La realización de esta práctica ha llevado una considerable cantidad de horas puesto que probar cada script es laborioso y en ciertos casos puedes dejar la maquina colgada, necesitando restaurarla completamente. Sin embargo, hemos aprendido una cantidad de bash scripting muy interesante e útil de cara al futuro. No ha resultado aburrida en ningún momento, solamente un poco tediosa a la hora de testearla. En general, una experiencia muy positiva.