

Automatización de tareas con Bash

1. Script de montaje de recursos CIFS y NFS (3 Puntos)

- Crear un script Bash que monte correctamente los recursos compartidos CIFS y NFS.
- Validar que los puntos de montaje existen y estén accesibles.



```
GNU nano 8.1 /etc/samba/smb.conf Modificado
path = /var/tmp
printable = Yes
create_mask = 0600
browseable = No

[print$]
comment = Printer Drivers
path = /var/lib/samba/drivers
write list = @printadmin root
force group = @printadmin
create_mask = 0664
directory mask = 0775

[compartido_cifs]
path = /srv/compartido_cifs
browsable = yes
writable = yes
guest ok = yes
read only = no
```

Primero creamos los archivos compartidos en el servidor para poder visualizarlos desde el cliente. Aquí se configura la carpeta `/srv/compartido_cifs` como un recurso compartido en Samba, permitiendo acceso de lectura y escritura sin restricciones de usuario (`guest ok`). Esta sección se añadió al final del archivo `/etc/samba/smb.conf`. La opción `browsable = yes` permite que la carpeta sea visible en el explorador de red desde equipos clientes.

```
GNU nano 5.6.1          root@localhost:~
montar_recursos.sh      Modificado

MONTAR_CIFS="/mnt/cifs"
MONTAR_NFS="/mnt/nfs"

# Usuario CIFS (ajústalo según tus credenciales reales)
USUARIO_CIFS="usuario"
CLAVE_CIFS="clave123"

# Crear puntos de montaje si no existen
mkdir -p "$MONTAR_CIFS"
mkdir -p "$MONTAR_NFS"

# Montar CIFS
echo "Montando CIFS..."
mount -t cifs "//$SERVIDOR_CIFS/compartido_cifs" "$MONTAR_CIFS" -o username=$USUARIO_CIFS,password=$CLAVE_CIFS

# Montar NFS
echo "Montando NFS..."
mount -t nfs "$SERVIDOR_NFS:/compartido_nfs" "$MONTAR_NFS"

# Verificar
echo "Montajes activos:"
mount | grep -E "$MONTAR_CIFS|$MONTAR_NFS"

^G Ayuda      ^O Guardar    ^W Buscar     ^K Cortar     ^T Ejecutar   ^C Ubicación  ^M-U Deshacer  ^M-A Poner marca
^X Salir      ^R Leer fich. ^N Reemplazar ^U Pegar      ^D Justificar ^_ Ir a línea  ^M-E Rehacer  ^M-G Copiar
```

Aquí creamos el script `montar_recursos.sh`. Se desarrolla un script en Bash que automatiza el montaje de recursos compartidos desde el servidor usando los protocolos CIFS y NFS. El script define rutas, credenciales y puntos de montaje, y ejecuta los comandos `mount` para cada protocolo. Además, valida y muestra si los montajes fueron exitosos utilizando un `grep` sobre el comando `mount`.

```
[root@localhost ~]# sudo ./montar_recursos.sh
Montando CIFS...
mount error(16): Device or resource busy
Refer to the mount.cifs(8) manual page (e.g. man mount.cifs) and kernel log messages (dmesg)
Montando NFS...
Montajes activos:
//192.168.46.151/compartido_cifs on /mnt/cifs type cifs (rw,relatime,vers=3.1.1,sec=none,cache=strict,upcall_target=app,uid=0,noforceuid,gid=0,noforcegid,addr=192.168.46.151,file_mode=0755,dir_mode=0755,soft,nounix,serverino,mapposix,repase=nfs,nativesocket,symlink=native,rsize=4194304,wsz=4194304,bsize=1048576,retrans=1,echo_interval=60,actimeo=1,closetimeo=1)
192.168.46.151:/srv/compartido_nfs on /mnt/nfs type nfs4 (rw,relatime,vers=4.2,rsize=262144,wsz=262144,namlen=255,hard,proto=tcp,timeo=600,retrans=2,sec=sys,clientaddr=192.168.46.152,local_lock=none,addr=192.168.46.151)
[root@localhost ~]#
```

Al ejecutar el script, el recurso NFS se monta correctamente, el script muestra los montajes activos, confirmando que el recurso NFS fue montado desde el servidor en `/mnt/nfs`. Se identifican los detalles técnicos del montaje como versión de protocolo, tamaños de buffers y dirección del cliente y servidor.

2. Script de backup automático con logging (4 Puntos)

- Crear un script que copie los archivos desde los recursos montados hacia un directorio local de backup.
- Incluir logs con marcas de tiempo que registren el éxito o falla de cada respaldo.

```

GNU nano 5.6.1 backup_automatico.sh Modificado
#!/bin/bash

# Script de respaldo automático desde recursos montados CIFS y NFS
# Cliente: 192.168.46.152

# Fecha actual con hora
FECHA=$(date +%F_%H-%M-%S)

# Carpetas de origen (montadas desde el servidor 192.168.46.151)
ORIGEN_CIFS="/mnt/cifs"
ORIGEN_NFS="/mnt/nfs"

# Carpeta de destino local
DESTINO="/backup/$FECHA"

# Archivo de log general
LOG="/backup/backup.log"

# Crear carpeta destino
mkdir -p "$DESTINO"

# Iniciar log con marca de tiempo
echo "===== Inicio de respaldo: $FECHA =====" >> "$LOG"
  
```

Se creó un script Bash en el cliente (192.168.46.152) que automatiza el respaldo de archivos montados desde el servidor. El script copia el contenido de las carpetas /mnt/cifs y /mnt/nfs a una ruta local /backup/ con fecha, y genera un log detallado. Esto permite asegurar trazabilidad y respaldo regular de los recursos compartidos.

```

[root@localhost ~]# nano backup_automatico.sh
[root@localhost ~]# chmod +x backup_automatico.sh
[root@localhost ~]# mkdir -p /backup
[root@localhost ~]# ./backup_automatico.sh
[root@localhost ~]# ls /backup
2025-08-05_11-26-18 backup.log
[root@localhost ~]# cat /backup/backup.log
===== Inicio de respaldo: 2025-08-05_11-26-18 =====
✓ CIFS montado, iniciando copia...
sending incremental file list
./

sent 58 bytes received 19 bytes 154,00 bytes/sec
total size is 0 speedup is 0,00
✓ NFS montado, iniciando copia...
sending incremental file list
./

sent 54 bytes received 19 bytes 146,00 bytes/sec
total size is 0 speedup is 0,00
✓ Respaldo finalizado: 2025-08-05_11-26-18

[root@localhost ~]#
  
```

Se ejecutó el script manualmente tras darle permisos y crear la carpeta /backup. El log muestra que los recursos CIFS y NFS estaban montados y que el respaldo se realizó correctamente, generando un registro con marca de tiempo. Esto demuestra el funcionamiento correcto del script y valida el cumplimiento del Requerimiento.

3. Automatización con cron y validación funcional (3 Puntos)

- Programar una tarea en cron que ejecute el script de backup diariamente.
- Probar el funcionamiento de la tarea y entregar evidencia de su ejecución (salida de cron, logs generados, etc.).

```

root@localhost:~# systemctl enable --now crond
root@localhost:~# systemctl status crond
● crond.service - Command Scheduler
   Loaded: loaded (/usr/lib/systemd/system/crond.service; enabled; preset: enabled)
   Active: active (running) since Tue 2025-08-05 10:19:04 -04; 1h 14min ago
     Main PID: 6541 (crond)
       Tasks: 1 (limit: 10721)
      Memory: 1.1M
         CPU: 74ms
    CGroup: /system.slice/crond.service
            └─6541 /usr/sbin/crond -n

ago 05 11:01:01 localhost.localdomain anacron[19506]: Will run job 'cron.daily' in 9 min.
ago 05 11:01:01 localhost.localdomain anacron[19506]: Will run job 'cron.weekly' in 29 min.
ago 05 11:01:01 localhost.localdomain anacron[19506]: Jobs will be executed sequentially
ago 05 11:01:01 localhost.localdomain run-parts[19508]: (/etc/cron.hourly) finished 0anacron
ago 05 11:01:01 localhost.localdomain CROND[19492]: (root) CMDEND (run-parts /etc/cron.hourly)
ago 05 11:10:01 localhost.localdomain anacron[19506]: Job 'cron.daily' started
ago 05 11:10:01 localhost.localdomain anacron[19506]: Job 'cron.daily' terminated
ago 05 11:30:01 localhost.localdomain anacron[19506]: Job 'cron.weekly' started
ago 05 11:30:01 localhost.localdomain anacron[19506]: Job 'cron.weekly' terminated
ago 05 11:30:01 localhost.localdomain anacron[19506]: Normal exit (2 jobs run)
root@localhost:~#

```

Primero verificamos que el servicio cron este activo. Se habilita y se inicia correctamente el servicio crond en el cliente (192.168.46.152), usando `systemctl enable --now crond`. Esto asegura que el sistema pueda ejecutar tareas programadas. El comando `status` confirma que el servicio está activo y funcionando correctamente en segundo plano.

```

root@localhost:~# cat /root/crontab
# Example of a crontab entry
# * * * * * /root/backup_automtico.sh
2:30 * * * * /root/backup_automtico.sh

```

Se edita el crontab del usuario root para programar la ejecución diaria del script `backup_automtico.sh` a las 2:30 AM. La línea añadida indica la ruta absoluta `/root/backup_automtico.sh`, obtenida previamente con `realpath`. Esto cumple con el requerimiento de automatizar el respaldo sin intervención manual.

```
root@localhost:~# crontab -e
no crontab for root - using an empty one
crontab: no changes made to crontab
root@localhost:~# crontab -e
no crontab for root - using an empty one
crontab: installing new crontab
root@localhost:~# crontab -l
30 2 * * * /root/backup_automatico.sh

root@localhost:~#
```

Se confirma que el crontab fue instalado correctamente y que la tarea aparece registrada mediante crontab -l. Esto valida que la automatización fue aplicada correctamente y que el sistema ejecutará el script en el horario establecido.

```
root@localhost:~# ls -l backup_automatico.sh
-rwxr-xr-x. 1 root root 1105 ago  5 11:25 backup_automatico.sh
root@localhost:~# realpath backup_automatico.sh
/root/backup_automatico.sh
root@localhost:~# bash /root/backup_automatico.sh
root@localhost:~# cat /backup/backup.log
===== Inicio de respaldo: 2025-08-05_11-26-18 =====
✓ CIFS montado, iniciando copia...
sending incremental file list
./

sent 58 bytes  received 19 bytes  154,00 bytes/sec
total size is 0  speedup is 0,00
✓ NFS montado, iniciando copia...
sending incremental file list
./

sent 54 bytes  received 19 bytes  146,00 bytes/sec
total size is 0  speedup is 0,00
✓ Respaldo finalizado: 2025-08-05_11-26-18

===== Inicio de respaldo: 2025-08-05_11-45-32 =====
✓ CIFS montado, iniciando copia...
sending incremental file list
./

sent 54 bytes  received 19 bytes  146,00 bytes/sec
total size is 0  speedup is 0,00
```

Se valida el funcionamiento correcto del script con dos ejecuciones manuales simulando la acción del cron. El archivo backup.log muestra entradas con marcas de tiempo distintas, confirmando que el script se ejecuta correctamente y realiza el respaldo desde CIFS y NFS. Esto prueba que la automatización funciona y genera logs auditables.