# Memoria de la Práctica
## Administración de Sistemas Informáticos
## *DATSI*

*Vidal Peña, Arturo*
w140307

24 de junio de 2021

# Índice

# 1. Introduccion

En este documento se detalla el proceso de realización del proyecto asignado. Dicho proyecto consiste en la realización de un script[1] en lenguaje Bash que sea capaz de instalar y configurar de forma autónoma un conjunto de servicios en un grupo de máquinas interconectadas en una red local.

## 1.1. Estructura de ficheros del proyecto

```
|-- README.md
|-- auxiliar
|   |-- backup_client.sh
|   |-- backup_server.sh
|   |-- common_functions.sh
|   |-- lvm.sh
|   |-- mount.sh
|   |-- nfs_client.sh
|   |-- nfs_server.sh
|   |-- nis_client.sh
|   |-- nis_server.sh
|   |-- raid.sh
|-- configs
|   |-- lvm.conf
|   |-- mount_raid.conf
|   |-- nfs_client.conf
|   |-- nfs_server.conf
|   |-- nis_client.conf
|   |-- raid.conf
|-- configurar_cluster.sh
|-- enunciado_asi.pdf
|-- memoria
|   |-- build.sh
|   |-- includes
|   |   |-- graph.png
|   |   |-- log.tex
|   |   |-- log.texr
|   |-- practica_asi.bcf
|   |-- practica_asi.bib
|   |-- practica_asi.pdf
|   |-- practica_asi.run.xml
|   |-- practica_asi.sty
|   |-- practica_asi.synctex.gz
|   |-- practica_asi.tex
|   |-- secciones
|       |-- backup.tex
|       |-- common_functions.tex
|       |-- dificultades.tex
|       |-- estructura_ficheros.tex
|       |-- estructura_ficheros.texr
```

```
|       |-- git_graph.tex
|       |-- introduccion.tex
|       |-- lvm.tex
|       |-- mount.tex
|       |-- nfs.tex
|       |-- nis.tex
|       |-- programa_principal.tex
|       |-- raid.tex
|       |-- timeline.tex
|-- pruebas
    |-- lvm
    |   |-- prueba_1.txt
    |-- mount
    |   |-- prueba_1.txt
    |-- raid
        |-- prueba_1.txt
        |-- prueba_test.txt
```

## 1.2. Funciones auxiliares

Aquí se encuentran definidas funciones auxiliares a los métodos principales del proyecto.

### 1.2.1. get_config_lines

Este método lee el fichero de configuración pasado como argumento y extrae cada una de sus líneas en una variable, de nombre `lines`, usada como array.

### 1.2.2. print_result

Recibe un valor numérico de retorno y la función desde la que se llamó al comando que devolvió dicho valor. Imprime un mensaje informando de éxito, o especificando la función en la que se dio error.

### 1.2.3. check_and_install_software

Este método recibe la dirección IP local y la de destino de la configuración, junto a un listado de los paquetes necesarios para ejecutar la función llamante.

Este método comprueba si ambas direcciones IP coinciden, en cuyo caso la instalación será local. En otro caso será sobre una máquina en remoto, para lo que necesitará instalar localmente (si no está previamente instalado) el paquete sshpass, para poder operar de manera cómoda en la máquina remota.

Después, comprobará si el paquete ya se encuentra instalado en la máquina objetivo, e instalará aquellos que no lo estén.

### 1.2.4. check_if_host_is_known

Este método comprueba si la IP pasada como argumento existe en el fichero `~/.ssh/known_hosts`. En caso de no existir, añade una nueva clave ssh para dicha IP.

#### 1.2.5. get_lvm_names_and_sizes

Esta función sólamente es llamada desde el método principal LVM.

Recibe una lista ordenada de los nombres y tamaños de los distintos volúmenes lógicos a crear, obtenidos del fichero de configuración.

Comprueba los distintos pares `<Nombre, Tamaño>` que encuentra y los separa de manera ordenada en dos listas diferentes, una para nombres y otra para tamaños, estando todos los pares en los mismos índices de ambas listas.

En el caso de que alguno no esté escrito adecuadamente, finaliza la ejecución con un error.

## 1.3. Programa principal

Lo primero que comprueba el programa principal es si usamos el número de argumentos correcto (uno, el fichero de configuración del cluster); o si usamos los argumentos de ayuda `-h, --help`, en cuyo caso nos imprime un mensaje de uso.

También comprueba si el fichero de configuración especificado existe y es un fichero (no un directorio, por ejemplo); y si estamos ejecutando el script como superusuario.

Una vez hechas todas las comprobaciones, lee el fichero de configuración y extrae la información del mismo (IP de destino, mandato y fichero de configuración para el mandato). Todo ello, siempre que cumpla con las normas de sintaxis del fichero, para lo que contrasta la línea contra una serie de expresiones regulares (RegEx), que filtrarán las líneas comentadas y avisará si alguna línea está mal escrita.

Si cumple las RegEx, llamará al servicio que se requiera para cada línea.

# 2. Configuración *Mount*

Esta función se encarga de montar el Sistema de Ficheros dado en el directorio de destino especificado, creándolo en caso de que no existiera.

Para ello, lo primero que hace es usar el método auxiliar `get_config_lines` para obtener las líneas del fichero de configuración, en las que aparece el nombre del sistema de ficheros a montar y del directorio de destino de montaje. Las líneas deberán ser de la siguiente manera:

1. Nombre del dispositivo
2. Directorio de montaje

Luego, hace uso de la función `check_and_install_software` para instalar, sea local o remotamente, los paquetes necesarios para realizar el montaje. Estos paquetes son:

- **Montaje local:**
  - mount
- **Montaje remoto:**
  - sshpass localmente
  - mount en el equipo de destino

Finalmente, crea el directorio de destino si no existía y luego monta el sistema de ficheros en él, haciendo uso del comando `mount`[2].

# 3. Configuración RAID

Esta función creará un sistema RAID en la máquina de destino especificada.
Las líneas del fichero de configuración del servicio se definen como sigue:

1. Nombre del dispositivo RAID
2. Nivel/Tipo de RAID
3. Dispositivos que lo componen

Después de obtener las líneas pertinentes del fichero de configuración, comprueba e instala los paquetes necesarios usando el método auxiliar `check_and_install_software`:

- Localmente:
  - mdadm
- En remoto:
  - sshpass localmente
  - mdadm en la máquina remota

Finalmente, usará el comando `mdadm`[4] en la máquina de destino (o mediante el comando `sshpass` en local) para crear el dispositivo RAID con el nombre, nivel y dispositivos especificados en el fichero de configuración.

# 4. Configuración LVM

Este método crea un Volumen Lógico en la máquina destino especificada.

Comprobará, a partir de la obtención de las líneas de configuración, que haya al menos tres líneas, puesto que el volumen lógico queda definido de la siguiente forma:

1. El nombre del volumen
2. Lista de dispositivos que lo componen
3. Nombre del volumen 1 y tamaño del mismo
4. Nombre del volumen 2 y tamaño del mismo
5. Nombre del volumen 3 y tamaño del mismo
6. etc.

En caso de no haber al menos tres líneas en el fichero de configuracíon, devolverá error.

Seguidamente, al igual que en los servicios anteriores, instalará el software necesario en las máquinas:

- Destino local:
  - pvcreate
  - vgcreate
  - lvcreate
- Destino remoto:
  - sshpass localmente
  - pvcreate en la máquina remota
  - vgcreate en la máquina remota
  - lvcreate en la máquina remota

También obtendrá, a partir de la tercera línea del fichero de configuración (y de las siguientes, si procediera), una lista con todos los nombres de los dispositivos y otra con todos los tamaños. Ambas listas están ordenadas de forma que compartan el índice, según la función auxiliar `get_lvm_names_and_sizes` definida anteriormente.

Finalmente, seguirá la siguiente secuencia:

1. Proceder a crear un grupo con la lista de dispositivos, con el comando `pvcreate`[3].
2. Crear el volumen lógico asociado a ese grupo, con el nombre especificado, usando el comando `vgcreate`[3].
3. Añadir los dispositivos y tamaños especificados, en orden, con el comando `lvcreate`[3].

# 5. Configuración NIS

# 6. Configuración NFS

# 7. Configuración *Backup*

# 8. Dificultades encontradas

Una de las primeras dificultades con las que nos encontramos fue la falta de tiempo que todos los miembros del equipo teníamos, debido en gran parte a estar compaginando estudios y trabajo. A pesar de ello, nos comprometimos en aportar todo lo posible para sacar adelante el proyecto.

Sin embargo, durante el transcurso del semestre, ninguno de los tres miembros aportamos lo suficiente para presentarla a tiempo. Aún así, siendo el que más aportó durante ese tiempo, decidí disolver el equipo y presentar el proyecto en solitario en la convocatoria extraordinaria.

# 9. Timeline del proyecto

A continuación se detalla el histórico de versiones del proyecto, de acuerdo con la historia extraída de la herramienta Git.

- ***Commit:*** **#ca64d28, Fecha: vie, 2 de octubre, Autor: Arturo Vidal Peña**

  Create README.md

- ***Commit:*** **#ffb75c0, Fecha: sáb, 3 de octubre, Autor: Arturo Vidal Peña**

  Estructura de la memoria

- ***Commit:*** **#12e689a, Fecha: jue, 8 de octubre, Autor: Javier Perez Martin**

  delete all comments and empty lines from the file

- ***Commit:*** **#3c91621, Fecha: dom, 11 de octubre, Autor: Javier Perez Martin**

  echo bad format line

- ***Commit:*** **#1597f55, Fecha: jue, 15 de octubre, Autor: Arturo Vidal Peña**

  Enunciado. Archivos de configuracion vacios. Funciones en script para facilitar desarrollo

- ***Commit:*** **#86f1ab1, Fecha: jue, 22 de octubre, Autor: Arturo Vidal Peña**

  Refactor functions in auxiliar files

- ***Commit:*** **#5786d34, Fecha: jue, 19 de noviembre, Autor: Arturo Vidal Peña**

  Primer intento mount

- ***Commit:*** **#e821ec9, Fecha: sáb, 21 de noviembre, Autor: Alfonso Sudara Padilla**

  lvm

- ***Commit:*** **#ce33532, Fecha: dom, 29 de noviembre, Autor: Arturo Vidal Peña**

  Mount funciona tanto en local como en host remoto. Añadido fichero de prueba.

- ***Commit:*** **#47df3c8, Fecha: dom, 6 de junio, Autor: Arturo Vidal Peña**

  Finished RAID configuration & tests.

- ***Commit:*** **#33bc6b6, Fecha: dom, 6 de junio, Autor: Arturo Vidal Peña**

  Started lvm. Initiated w/ example test & config

- ***Commit:*** **#e562c2a, Fecha: mar, 8 de junio, Autor: Arturo Vidal Peña**

  Completed lvm for both local & remote execution.

- ***Commit:*** **#5e2c6da, Fecha: mar, 8 de junio, Autor: Arturo Vidal Peña**

  Added documentation to common functions.
- ***Commit:*** **#8a57973, Fecha: mar, 8 de junio, Autor: Arturo Vidal Peña**

  Fixed typo. Added package installation call.
- ***Commit:*** **#e70d20b, Fecha: dom, 20 de junio, Autor: Arturo Vidal Peña**

  Updated gitignore to ommit LaTeX auxiliar files.
- ***Commit:*** **#fc13d49, Fecha: dom, 20 de junio, Autor: Arturo Vidal Peña**

  Removed partners from proyect memoir.
- ***Commit:*** **#4b70c78, Fecha: mar, 22 de junio, Autor: Arturo Vidal Peña**

  Added Bash script to automate git log and graph info recopilation & write it into LaTeX files.
- ***Commit:*** **#158845d, Fecha: mar, 22 de junio, Autor: Arturo Vidal Peña**

  Initial git timeline.
- ***Commit:*** **#85ef664, Fecha: mar, 22 de junio, Autor: Arturo Vidal Peña**

  Initial git graph.
- ***Commit:*** **#cc3137a, Fecha: mar, 22 de junio, Autor: Arturo Vidal Peña**

  Initial difficulties statement.
- ***Commit:*** **#a8538ff, Fecha: mar, 22 de junio, Autor: Arturo Vidal Peña**

  Introduction, main program & common functions explained.
- ***Commit:*** **#4d92d71, Fecha: mar, 22 de junio, Autor: Arturo Vidal Peña**

  Added git-graph directory to gitignore as it's not needed.
- ***Commit:*** **#4a9ae93, Fecha: mar, 22 de junio, Autor: Arturo Vidal Peña**

  Added includes dir with git-graph image & git log file.
- ***Commit:*** **#0666a92, Fecha: mar, 22 de junio, Autor: Arturo Vidal Peña**

  Renamed functions LaTeX files.
- ***Commit:*** **#f92df34, Fecha: mar, 22 de junio, Autor: Arturo Vidal Peña**

  Updated main LaTeX files with current file names.
- ***Commit:*** **#16d7fe2, Fecha: mié, 23 de junio, Autor: Arturo Vidal Peña**

  Updated gitignore to ommit LaTeX auxiliar files & accept correct ones.
- ***Commit:*** **#4744ffe, Fecha: mié, 23 de junio, Autor: Arturo Vidal Peña**

Updated gitignore to ommit LaTeX auxiliar files & accept correct ones.

- ***Commit:*** **#e563579, Fecha: jue, 24 de junio, Autor: Arturo Vidal Peña**

  Updated gitignore to ommit LaTeX auxiliar files & accept correct ones.
- ***Commit:*** **#cb7d9dd, Fecha: jue, 24 de junio, Autor: Arturo Vidal Peña**

  Added LVM explanation.
- ***Commit:*** **#ba95d6e, Fecha: jue, 24 de junio, Autor: Arturo Vidal Peña**

  Added introduction explanation.
- ***Commit:*** **#22973b2, Fecha: jue, 24 de junio, Autor: Arturo Vidal Peña**

  Added mount explanation.
- ***Commit:*** **#0db6a3b, Fecha: jue, 24 de junio, Autor: Arturo Vidal Peña**

  Added raid explanation.
- ***Commit:*** **#67b90bd, Fecha: jue, 24 de junio, Autor: Arturo Vidal Peña**

  Added main program explanation.
- ***Commit:*** **#b771985, Fecha: jue, 24 de junio, Autor: Arturo Vidal Peña**

  Added bibliography.

# Bibliografía


[1] Gabriel Cebrián Márquez. *Administración de sistemas UNIX - Proyecto práctico. Script maestro para la configuración de un cluster Linux.*

[2] ManPages. *Debian Manpages - mount(8) - mount.* URL: `https://manpages.debian.org/testing/mount/mount.8.en.html`. (accessed: 29.11.2020).

[3] Debian Wiki. *Debian - Configure LVM.* URL: `https://wiki.debian.org/es/LVM`. (accessed: 08.06.2021).

[4] Server World. *CentOS 7 - Configure RAID 1.* URL: `https://www.server-world.info/en/note?os=CentOS_7&p=raid1`. (accessed: 06.06.2021).