



Computación Aplicada

TRABAJO PRACTICO INTEGRADOR

Alumnos:

Flores Valentin

Camarotta Lucas

Armendariz Iñaki

Trabajo práctico integrador grupal

Consignas

Condiciones generales

- a) Este trabajo práctico se debe realizar en una máquina virtual con **GNU/Linux Debian preinstalado**, que se encuentra disponible, para su descarga, en **Blackboard**. La máquina virtual ya está configurada específicamente para esta actividad, y debe importarse como un sistema virtualizado.
Las instrucciones para la importación se encuentran en el sitio web de Oracle VirtualBox.
- b) **Grupos:** los equipos de trabajo **no deben exceder los 5 integrantes**, y deben estar **registrados en Blackboard**.
- c) **Calificación:** para aprobar, es necesario tener el 60% del trabajo práctico bien y funcionando. Asimismo, se deberá realizar la defensa oral en modalidad sincrónica sobre las tareas realizadas.

1) Configuración del entorno

- 1) La máquina virtual está dividida en partes y comprimidas en formato **“.rar”**.
Descargar y ensamblar los archivos utilizando herramientas como WinRar.
- 2) La clave de **root** es desconocida inicialmente, por lo que deberá ser blanqueada antes de comenzar. Una vez dentro del sistema operativo, la misma debe ser cambiada por **“palermo”** (sin comillas).
- 3) Establecer el nombre de *hostname* como **TPServer**.

Para comenzar reestablecemos la contraseña del usuario root.

Se editan los parámetros del GRUB (presionando la tecla ‘e’) para iniciar una consola con permisos root.

```
TPVMCA [Running] - Oracle VirtualBox

GNU GRUB  versión 2.06-3~deb11u6

set root='hd0,msdos1'
if [ x$feature_platform_search_hint = xy ]; then
  search --no-floppy --fs-uuid --set=root --hint-bios=hd0,msdos1\
--hint-efi=hd0,msdos1 --hint-baremetal=ahci0,msdos1  cd2cb4ca-39bd-4522\
-919c-e4ceb81b5ae8
else
  search --no-floppy --fs-uuid --set=root cd2cb4ca-39bd-4522-919\
c-e4ceb81b5ae8
fi
echo          'Loading Linux 5.10.0-30-amd64 ...'
linux        /boot/vmlinuz-5.10.0-30-amd64 root=UUID=cd2cb4ca-39\
bd-4522-919c-e4ceb81b5ae8 rw init=/bin/bash quiet
echo          'Loading initial ramdisk ...'
initrd       /boot/initrd.img-5.10.0-30-amd64

Se soporta una edición de pantalla mínima al estilo de Emacs.
«TAB» enumera las posibles palabras a completar. Pulse «Ctrl-x»
o «F10» para arrancar, «Ctrl-c» o «F2» para una línea de órdenes
o «ESC» para descartar las ediciones y volver al menú de GRUB.
```

Luego se monta el sistema en modo lectura/escritura y con passwd se reestablece la contraseña.

```
mount -o remount,rw /
passwd root
exec /sbin/init
```

Al reiniciar el sistema se editan los archivos `/etc/hosts` y `/etc/hostname` para configurar el hostname TPServer

```
TPVMCA [Running] - Oracle VirtualBox

Debian GNU/Linux 11 TPServer tty1

TPServer login: root
Password:
Linux TPServer 5.10.0-30-amd64 #1 SMP Debian 5.10.218-1 (2024-06-01) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Nov  9 02:19:47 -03 2025 on tty1
root@TPServer:~# cat /etc/hosts
127.0.0.1    localhost
127.0.1.1    TPServer

# The following lines are desirable for IPv6 capable hosts
::1        localhost ip6-localhost ip6-loopback
ff02::1    ip6-allnodes
ff02::2    ip6-allrouters
root@TPServer:~# cat /etc/hostname
TPServer
root@TPServer:~# hostname
TPServer
root@TPServer:~#
```

2) Servicios

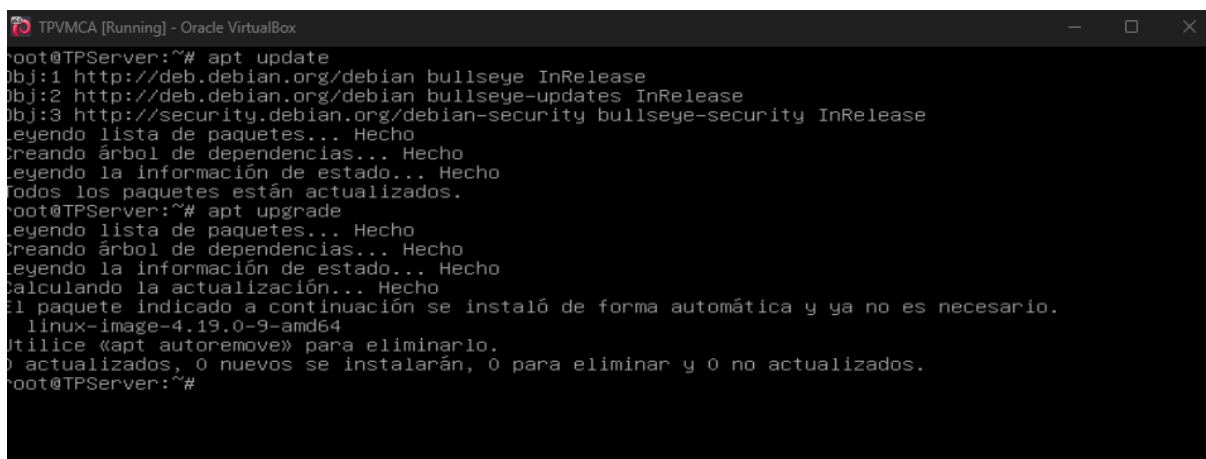
- 1) **SSH:** instalar y configurar el servicio de **SSH**. El servidor debe permitir el acceso al usuario *root* mediante una clave privada/pública, proporcionada junto con la máquina virtual en **Blackboard**.
- 2) **WEB:** instalar y configurar el servidor **Apache** con soporte para PHP (versión 7.3 o superior). Configurar el servidor para servir el archivo “index.php” y “logo.png”, disponible junto con la máquina virtual en **Blackboard**.
- 3) **Base de datos:** instalar y configurar **MariaDB**. Cargar, en el motor de base de datos, el script SQL, denominado “db.sql”, disponible junto con la máquina virtual en **Blackboard**.

Nota: las pruebas de conectividad y acceso al sitio web deben realizarse desde una máquina física u otra máquina en la misma red.

Previo a instalar nuevos paquetes actualizamos el sistema corriendo los comandos

```
apt update
apt upgrade
```

Hubo un problema con los repositorios preconfigurados en */etc/apt/sources.list* por lo que los modificamos para usar <http://deb.debian.org/debian>



```
TPVMCA [Running] - Oracle VirtualBox
root@TPServer:~# apt update
Obj:1 http://deb.debian.org/debian bullseye InRelease
Obj:2 http://deb.debian.org/debian bullseye-updates InRelease
Obj:3 http://security.debian.org/debian-security bullseye-security InRelease
leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
leyendo la información de estado... Hecho
Todos los paquetes están actualizados.
root@TPServer:~# apt upgrade
leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
leyendo la información de estado... Hecho
Calculando la actualización... Hecho
El paquete indicado a continuación se instaló de forma automática y ya no es necesario.
linux-image-4.19.0-9-amd64
Utilice «apt autoremove» para eliminarlo.
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
root@TPServer:~#
```

Instalación servicio ssh

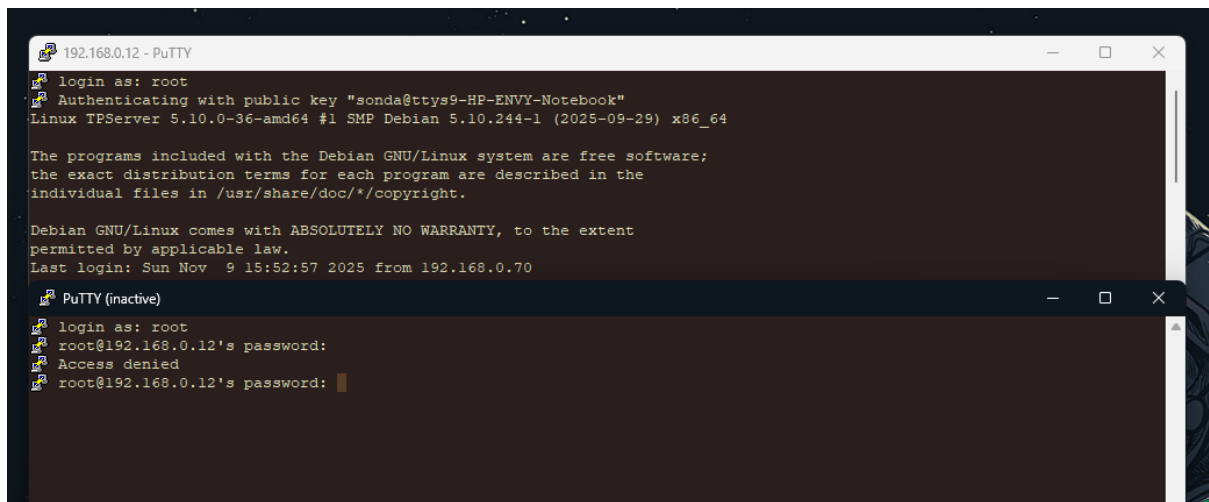
```
apt install openssh-server
```

Copiamos la clave publica a `/root/.ssh/authorized_keys`

Habilitamos el login del usuario root editando el archivo `/etc/ssh/sshd_config` permitiendo la línea `PermitRootLogin prohibit-password`

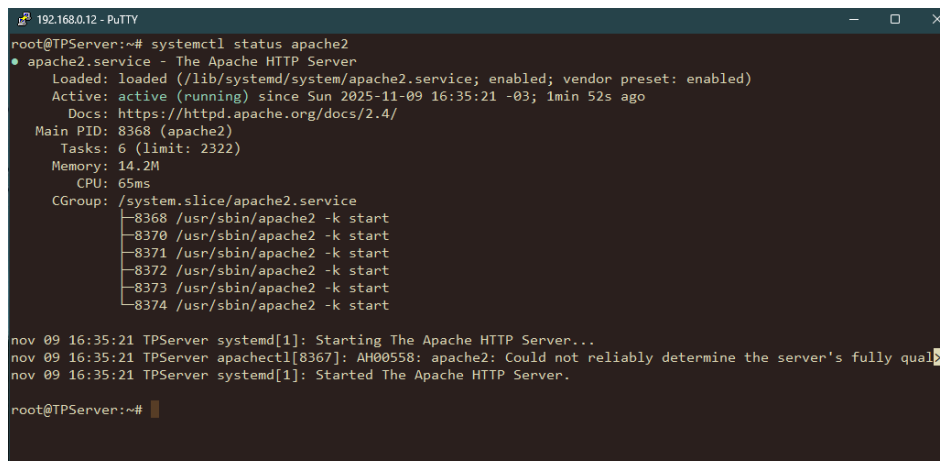
Comprobamos que no funciona el login por ssh utilizando usuario y clave.

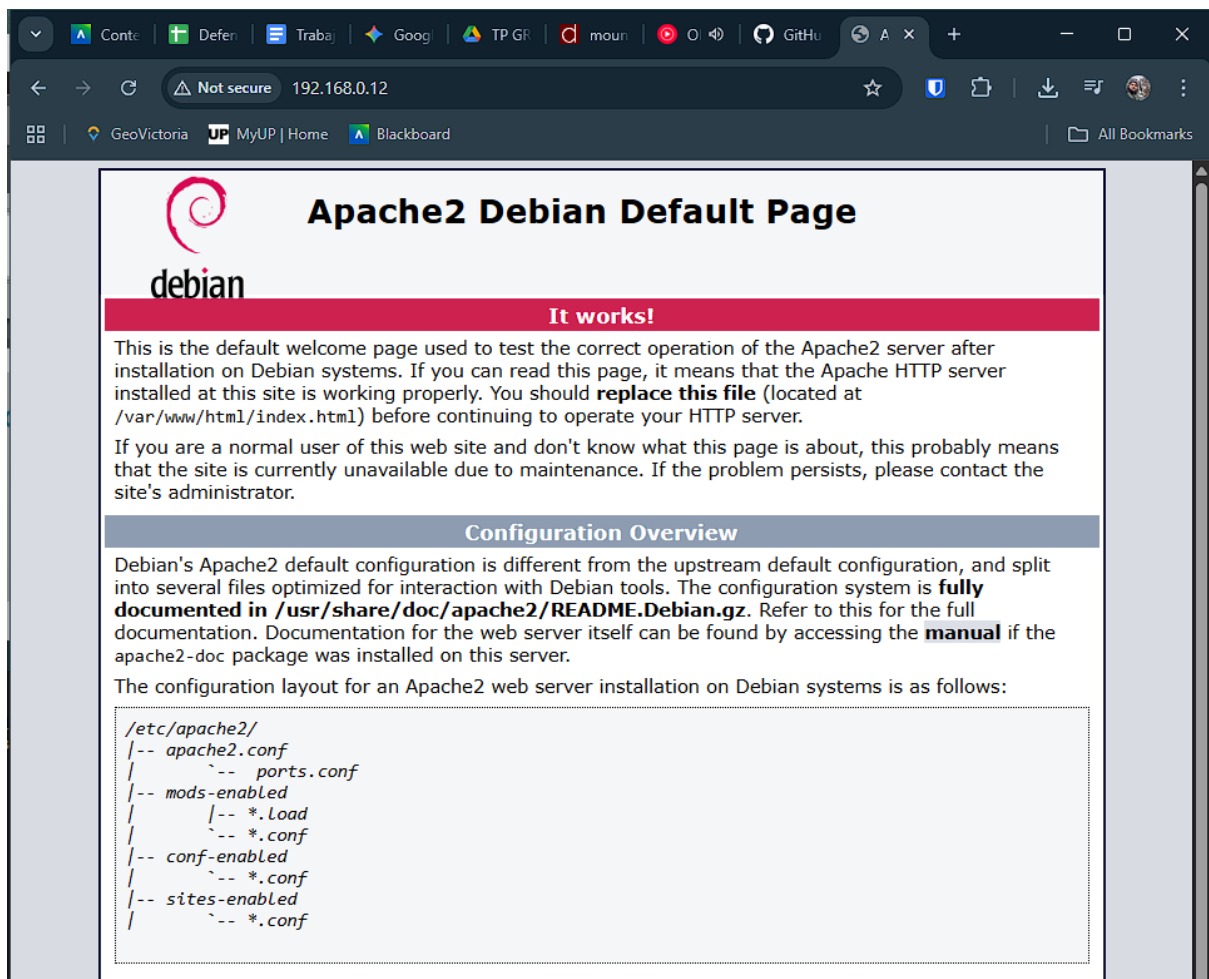
Si funciona utilizando la clave privada.



Instalación apache server

```
apt install apache2 php libapache2-mod-php
systemctl enable apache2
systemctl start apache2
```





Instalación MariaDB

```
apt install mariadb-server
mysql_secure_installation
mysql -u root -p < /tmp/db.sql
```

Instalamos MariaDB y corrimos el script de configuración segura.
Luego corrimos el script indicado para el ejercicio.

```
192.168.0.12 - PuTTY
root@TPServer:~# mysql
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 42
Server version: 10.5.29-MariaDB-0+deb11u1 Debian 11

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
MariaDB [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| ingenieria |
| mysql |
| performance_schema |
+-----+
4 rows in set (0,001 sec)

MariaDB [(none)]>
```

3) Configuración de Red

- 1) Configurar la interfaz de red con una IP estática en el archivo de configuración. La IP debe pertenecer al mismo rango red de la máquina física.
- 2) El archivo de configuración debe incluir los campos **ADDRESS**, **NETMASK** y **GATEWAY**.

```
192.168.0.2 - PuTTY
root@TPServer:~# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:1f:b4:4b brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.2/24 brd 192.168.0.255 scope global enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe1f:b44b/64 scope link
        valid_lft forever preferred_lft forever
root@TPServer:~# cat /etc/network/interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
    address 192.168.0.2
    netmask 255.255.255.0
    gateway 192.168.0.1
root@TPServer:~#
```

4) Almacenamiento

- 1) Agregar un nuevo disco de **10 GB** adicional a la máquina virtual.
- 2) Crear dos particiones estándar (tipo 83), con las siguientes capacidades:
 - **/www_dir: 3 GB**
 - **/backup_dir: 6 GB**
- 3) Configurar el directorio **/www_dir** para alojar el archivo **index.php** y **logo.png**. Actualizar el archivo de configuración de Apache para que éste apunte a la nueva ubicación (ver archivos 000-default.conf y apache2).
- 4) Configurar el directorio **/www_dir** para que se monte automáticamente al iniciar el sistema operativo.
- 5) Configurar el directorio **/backup_dir** para que se monte automáticamente al iniciar el sistema operativo.

Nota: se debe crear un archivo en /opt llamado "particion", y redirigir el contenido del archivo "partitions" ubicado en /proc (el archivo original es efímero y se pierde al apagar la máquina).

El siguiente paso fue crear los puntos de montaje con mkdir

```
vflores@vaflores:~$ sudo mkfs.ext4 /dev/nvme0n1p2
mke2fs 1.46.2 (28-Feb-2021)
Creating filesystem with 1572864 4k blocks and 393216 inodes
Filesystem UUID: 2e69dd75-f4ee-4212-9279-6ea5f8bc10dc
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done
```

Se usó el comando **blkid** para obtener los UUID (identificadores únicos de partición).

```
vflores@vaflores:~$ sudo blkid /dev/nvme0n1p1
/dev/nvme0n1p1: UUID="31159369-b537-4a44-9bd1-ba5f3f05e75c" BLOCK_SIZE="4096" TY
PE="ext4" PARTUUID="9caeb9ba-01"
vflores@vaflores:~$ sudo blkid /dev/nvme0n1p2
/dev/nvme0n1p2: UUID="2e69dd75-f4ee-4212-9279-6ea5f8bc10dc" BLOCK_SIZE="4096" TY
PE="ext4" PARTUUID="9caeb9ba-02"
```


Posteriormente, se editó el archivo `/etc/fstab` con nano, añadiendo dos líneas. Cada línea incluía el UUID de la partición, el punto de montaje, el tipo de sistema de archivos (ext4) y las opciones predeterminadas.

```
vflores@vaflores:~$ sudo nano /etc/fstab
vflores@vaflores:~$ sudo mount -a
vflores@vaflores:~$ df -h | grep -E 'www_dir|backup_dir'
/dev/nvme0n1p1 2.9G 24K 2.8G 1% /www_dir
/dev/nvme0n1p2 5.9G 24K 5.6G 1% /backup_dir
vflores@vaflores:~$
```

Finalmente, confirmamos el correcto funcionamiento de las particiones al ejecutar `mount -a`, que se encarga de montar todas las particiones definidas en `/etc/fstab`. Posteriormente, utilizamos `df -h` para visualizar el estado del espacio en disco, lo que nos permitió verificar que ambas particiones se encontraban activas y correctamente montadas.

```
vflores@vaflores:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            962M   0 962M   0% /dev
tmpfs           198M  1.2M  196M   1% /run
/dev/sda1        19G   4.7G   13G  27% /
tmpfs           986M   0 986M   0% /dev/shm
tmpfs           5.0M   4.0K   5.0M   1% /run/lock
tmpfs           198M  112K  197M   1% /run/user/1000
/dev/nvme0n1p1  2.9G   28K   2.8G   1% /www_dir
/dev/nvme0n1p2  5.9G   24K   5.6G   1% /backup_dir
```

Gracias a este procedimiento, se configuró el servidor Apache para servir archivos desde `/www_dir`. Asimismo, se utilizó el directorio `/backup_dir` como destino para los archivos generados por el script de backup automático.

Movimos los archivos `index.php` y `logo.png` a `/www_dir` y asignamos los permisos al usuario de apache.

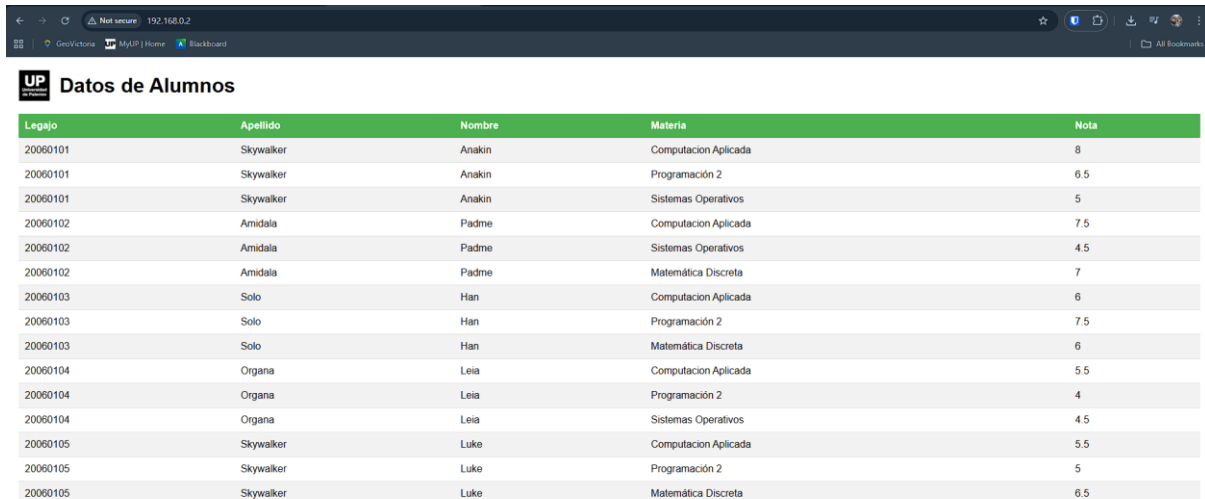
```
vflores@vaflores:~$ sudo chown -R www-data:www-data /www_dir | sudo chmod -R 755 /www_dir
```

Editamos `/etc/apache2/sites-available/000-default.conf` donde se modifica `'DocumentRoot /var/www/html'` por `'DocumentRoot /www_dir'`

Editamos `/etc/apache2/apache2.conf` para agregar:

```
<Directory /www_dir>
    Options Indexes FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
```

El servidor web funciona correctamente tras instalar: `apt install php-mysql`



The screenshot shows a web browser window with the address bar displaying "192.168.0.2". The page title is "UP Datos de Alumnos". The table contains student records with columns: Legajo, Apellido, Nombre, Materia, and Nota.

Legajo	Apellido	Nombre	Materia	Nota
20060101	Skywalker	Anakin	Computacion Aplicada	8
20060101	Skywalker	Anakin	Programación 2	6.5
20060101	Skywalker	Anakin	Sistemas Operativos	5
20060102	Amidala	Padme	Computacion Aplicada	7.5
20060102	Amidala	Padme	Sistemas Operativos	4.5
20060102	Amidala	Padme	Matemática Discreta	7
20060103	Solo	Han	Computacion Aplicada	6
20060103	Solo	Han	Programación 2	7.5
20060103	Solo	Han	Matemática Discreta	6
20060104	Organa	Leia	Computacion Aplicada	5.5
20060104	Organa	Leia	Programación 2	4
20060104	Organa	Leia	Sistemas Operativos	4.5
20060105	Skywalker	Luke	Computacion Aplicada	5.5
20060105	Skywalker	Luke	Programación 2	5
20060105	Skywalker	Luke	Matemática Discreta	6.5

Nota: se debe crear un archivo en `/opt` llamado "particion", y redirigir el contenido del archivo "partitions" ubicado en `/proc` (el archivo original es efímero y se pierde al apagar la máquina).

```
vflores@vaflores:~$ sudo cp /proc/partitions /opt/particion
vflores@vaflores:~$ ls -l /opt/particion
-r--r--r-- 1 root root 279 Nov  8 14:54 /opt/particion
vflores@vaflores:~$ cat /opt/particion
major minor #blocks name
259      0    10485760 nvme0n1
259      3     3145728 nvme0n1p1
259      4     6291456 nvme0n1p2
11       0     1048575 sr0
8        0    20971520 sda
8        1    19970048 sda1
8        2         1 sda2
8        5     998400 sda5
```

5) Backup

- 1) Desarrollar un script de *backup* denominado “**backup_full.sh**”, y guardarlo en **/opt/scripts**.
- 2) El script debe backupear los directorios indicados con nombres que incluyan la fecha en formato **ANSI (YYYYMMDD)**. Por ejemplo, para **/var/log**, el archivo generado debería llamarse “**log_bkp_20240302.tar.gz**”.
- 3) Los backups generados deben almacenarse en la partición que tiene montado el directorio **/backup_dir**.
- 4) El script debe aceptar argumentos como **origen** (lo que se va a backupear) y **destino** (dónde se va a backupear).
- 5) El script debe incluir una opción de ayuda (**-help**), para guiar al usuario en el uso del script.
- 6) El script **debe validar** que los sistemas de archivos de **origen** y **destino** estén disponibles antes de ejecutar el backup.
- 7) El script debe ser incluido en un calendario de tareas para correr automáticamente:
 - **TODOS LOS DÍAS a las 00:00 hs: Backupear “/var/logs”**
 - **LUNES, MIÉRCOLES, VIERNES a las 23:00 hs: Backupear “/www_dir”**

Para este punto, creamos un script de backup automatizado llamado `backup_full.sh`, ubicado en `/opt/scripts`.

Este script está diseñado para generar copias de seguridad comprimidas de varios directorios del sistema. Los nombres de los archivos de backup incluyen la fecha en formato ANSI (AAAAMMDD) y se almacenan en `/backup_dir`, un directorio montado en una partición independiente de 6GB.

El script acepta argumentos de origen y destino, incluye una opción `-help` para explicar su uso, y realiza validaciones para asegurar que los directorios especificados existan antes de ejecutar la copia de seguridad. Esto garantiza una automatización segura de las tareas programadas.

```

# -----
# FUNCION DE AYUDA
# -----
show_help() {
    echo "Uso: $0 [OPCION] [ORIGEN] [DESTINO]"
    echo ""
    echo "Realiza un backup de un directorio ORIGEN al directorio DESTINO, "
    echo "nombrando el archivo con la fecha en formato YYYYMMDD."
    echo ""
    echo "Opciones disponibles:"
    echo "  -help      Muestra esta ayuda."
    echo ""
    echo "Argumentos obligatorios:"
    echo "  ORIGEN     Ruta absoluta del directorio a backupear (ej: /var/log)"
    echo "  DESTINO     Ruta absoluta del directorio donde se guardará el backup (ej: /b"
    echo ""
    echo "Ejemplo: $0 /var/log /backup_dir"
}

```

```

# -----
# FUNCION DE VALIDACION DE MONTAJE
# -----
check_mount() {
    local dir=$1
    if mountpoint -q "$dir"; then
        echo "Validación OK: El directorio '$dir' está montado correctamente." | tee ->
        return 0
    else
        echo "Error de Validación: El directorio '$dir' NO está montado o no existe." >
        return 1
    fi
}

```

```

# -----
# MANEJO DE ARGUMENTOS
# -----

# Opción de ayuda
if [ "$1" == "-help" ]; then
    show_help
    exit 0
fi

# Validar número de argumentos
if [ "$#" -ne 2 ]; then
    echo "Error: Se requieren dos argumentos (ORIGEN y DESTINO)." | tee -a "$LOG_FILE"
    show_help
    exit 1
fi

SOURCE_DIR=$1
DEST_DIR=$2

# Validar que el ORIGEN sea un directorio
if [ ! -d "$SOURCE_DIR" ]; then
    echo "Error: EL ORIGEN ($SOURCE_DIR) no es un directorio válido." | tee -a "$LOG_F
    exit 1
fi

```

```

# EJECUCION DEL BACKUP
# -----

# 1. Validar si los sistemas de archivos están disponibles
if check_mount "$SOURCE_DIR" && check_mount "$DEST_DIR"; then
    # 2. Definir el nombre del archivo de backup
    BASE_NAME=$(basename "$SOURCE_DIR")
    ARCHIVE_NAME="${BASE_NAME}_bkp_${DATE_ANSI}.tar.gz"
    FULL_PATH="${DEST_DIR}/${ARCHIVE_NAME}"

    echo "--- Iniciando backup de $SOURCE_DIR a $FULL_PATH ---" | tee -a "$LOG_FILE"

    # 3. Crear el backup
    if tar -czf "$FULL_PATH" -C "$(dirname "$SOURCE_DIR")" "$BASE_NAME"; then
        echo "Backup exitoso: $ARCHIVE_NAME" | tee -a "$LOG_FILE"
    else
        echo "Error: Falló la creación del backup TAR." | tee -a "$LOG_FILE"
        exit 2
    fi
else
    echo "Error: El backup no se ejecutó debido a fallos en la validación de montaje."
    exit 3
fi

exit 0

```

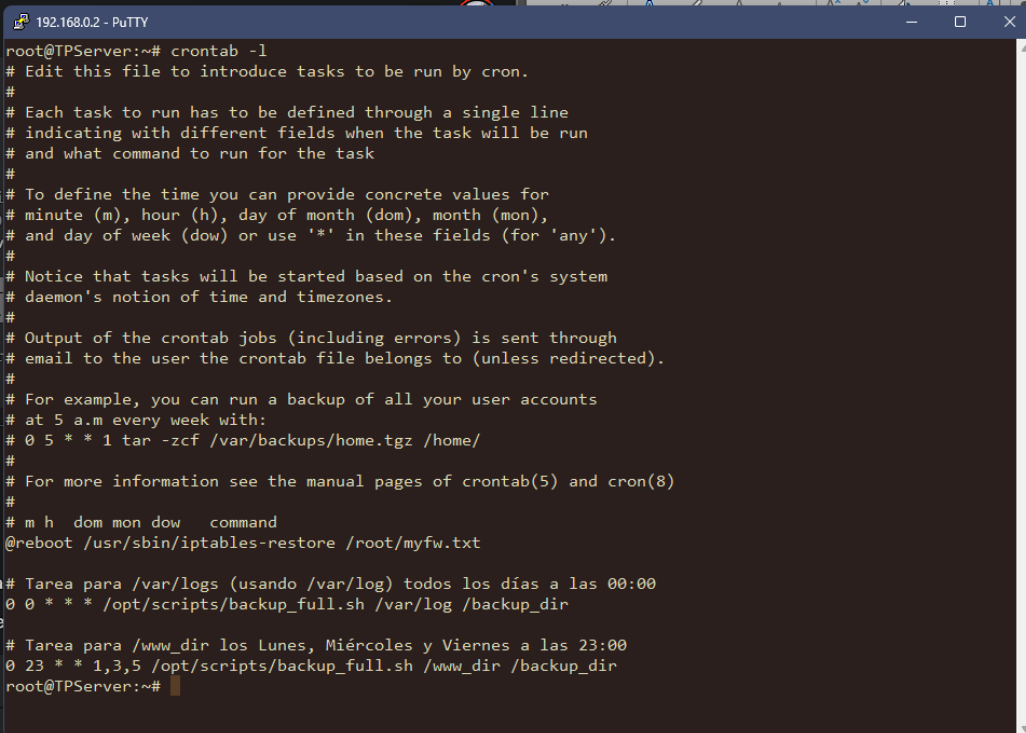
```
vflores@vaflores:~$ sudo /opt/scripts/backup_full.sh /www_dir /backup_dir
Validación OK: El directorio '/www_dir' está montado correctamente.
Validación OK: El directorio '/backup_dir' está montado correctamente.
--- Iniciando backup de /www_dir a /backup_dir/www_dir_bkp_20251108.tar.gz ---
Backup exitoso: www_dir_bkp_20251108.tar.gz
```

Programación de tarea:

```
crontab -e
```

```
# Tarea para /var/logs todos los días a las 00:00
0 0 * * * /opt/scripts/backup_full.sh /var/log /backup_dir
```

```
# Tarea para /www_dir los Lunes, Miércoles y Viernes a las 23:00
0 23 * * 1,3,5 /opt/scripts/backup_full.sh /www_dir /backup_dir
```



```
root@TPServer:~# crontab -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
@reboot /usr/sbin/iptables-restore /root/myfw.txt

# Tarea para /var/logs (usando /var/log) todos los días a las 00:00
0 0 * * * /opt/scripts/backup_full.sh /var/log /backup_dir
#
# Tarea para /www_dir los Lunes, Miércoles y Viernes a las 23:00
0 23 * * 1,3,5 /opt/scripts/backup_full.sh /www_dir /backup_dir
root@TPServer:~#
```

6) Entregables

- 1) Crear un repositorio en **github (es gratuito)**, y redactar el README.md con los nombres de los participantes del grupo.
- 2) Subir, en el repositorio creado, los directorios:
 - **“/root”, “/etc”, “/opt”, “/www_dir” y “/backup_dir”. Todos ellos comprimidos individualmente en formato “.tar.gz”.**
 - **“/var”** se lo debe splitear en partes pequeñas para que pueda ser subido.
- 3) Realizar un diagrama topológico de la infraestructura armada.

El diagrama topológico solicitado en este punto lo subimos al repositorio como imagen png.