


MANDATORY

1. sudo

 **Que es sudo ?** Is a program that enables a user to run commands as other user (root by default) getting access to all the privileges the other user has.

Login as root

```
$ su -
```

Install sudo and configure

```
$ apt install sudo
```


Check to verify

```
$ sudo -V | more
```

Or

```
$ dpkg -l | grep sudo
```

2. SSH

 **Que es SSH ?** Es el nombre de un protocolo y del programa que lo implementa cuya principal función es el acceso remoto a un servidor por medio de un canal seguro en el que toda la información está cifrada.

Install

```
$ sudo apt update
$ sudo apt install openssh-server
```

Check:

```
$ dpkg -l | grep ssh
```

Configuring

```
$ sudo nano /etc/ssh/sshd_config
#Port 22 → Port 4242
#PermitRootLogin prohibit-password → PermitRootLogin no
```

```
$ sudo nano /etc/ssh/ssh_config
#Port 22 → Port 4242
```

Restart the SSH service

```
$ sudo service ssh restart
```

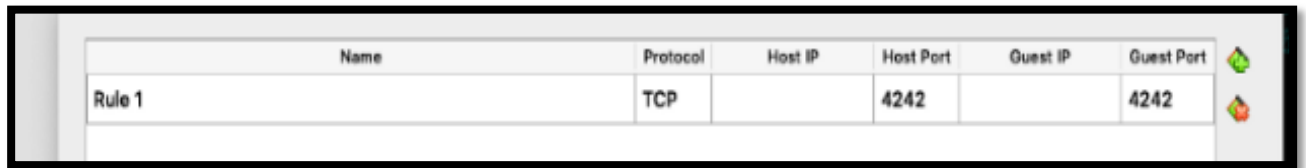
Check the SSH server status

```
$ sudo service ssh status
```

3. Connecting SSH server

Add forward rule for VirtualBox

1. Go to VirtualBox-> Choose the VM->Select Settings
2. Choose "Network" -> "Adapter 1"->"Advanced"->"Port Forwarding"
3. Enter the values as shown:



Name	Protocol	Host IP	Host Port	Guest IP	Guest Port
Rule 1	TCP		4242		4242

4. From host side from iTerm2 or Terminal enter as shown below:

```
$ ssh your_username@localhost -p 4242
```

5. Quit the connection:

```
$ exit
```

4. UFW (Uncomplicated Firewall)

🧠 **Que es UFW ?** Es un [firewall](#) el cual utiliza la línea de comandos para configurar las "[iptables](#)" usando un pequeño número de comandos simples.

Install UFW

```
$ sudo apt install ufw
```

Check:

```
$ dpkg -l | grep ufw
```

Enable

```
$ sudo ufw enable
```

Configure the port rules

```
$ sudo ufw allow 4242
```

Check the status

```
$ sudo ufw status
```

Delete the new rule: (This is for when you defend your Born2beroot)

```
$ sudo ufw status numbered  
$ sudo ufw delete (that number, for example 5 or 6)
```

5. Password policy

This setting enforces how many classes, i.e upper-case, lower-case, and other characters, should be in a password, also the length of the password.

Installing password quality checking library (libpam-pwquality):

```
$ sudo apt install libpam-pwquality
```

Change the policy

```
$ sudo nano /etc/pam.d/common-password
```

Find this line: Password requisite pam_pwquality.so retry=3 and add

minlen=10	Cantidad mínima de caracteres
uccredit=-1	Como mínimo (-) un carácter. +1 sería como máximo
dccredit=-1	Como mínimo un dígito
maxrepeat=3	No puede repetirse un carácter más de 3 veces
reject_username	No puede contener el nombre de usuario (también <code>usercheck=1</code>)
difok=7	7 caracteres que no formen parte de la antigua pwd
enforce_for_root	Implementamos esta política para root

Password expiration:

```
$ sudo nano /etc/login.defs
```

Find this part and change it like this:(max 30 days, min number of days (2) allowed before the modification, receive a notification before expiration at least 7 days before)

```
PASS_MAX_DAYS 9999 ---> PASS_MAX_DAYS 30
PASS_MIN_DAYS 0 ---> PASS_MIN_DAYS 2
```

Before making any other changes, you must run these commands.

```
chage <yourlogin>
  Minimum Password Age [0]: 2
  Maximum Password Age [9999]: 30
  Last Password Change (YYYY-MM-DD) [date]: <last time you change password>
  Password Expiration Warning [7]: 7
  Enter for the rest)

chage root
  <Same as before>
```


Reboot the change affects:

```
$ sudo reboot
```

6. Users and groups

Creamos los grupos user42 y evaluating (durante la defensa)

```
$ sudo addgroup user42
$ sudo addgroup evaluating
```

 **Que es GID ?** Es el identificador de grupo, es una abreviatura de Group **ID**.

Check if group created:

```
$ getent group (group name)
      o
$ cat /etc/group
```

Check the all local users:

```
$ cut -d: -f1 /etc/passwd
```

Create the user

```
$ sudo adduser new_username
```

Check if the user was successfully created

```
$ getent passwd user_name
```

Assign an user into a group

```
$ sudo adduser your_username group_name
```

Or

```
$ sudo usermod -aG group_name username
```

Check if the users are in group

```
$ getent group group_name
```

Check which groups user account belongs:

```
$ sudo groups (user)
```

Check if password rules working in users:

```
$ chage -l your_new_username
```

7. Configuring sudo group

Create and edit files:

```
$ mkdir /var/log/sudo
$ touch /var/log/sudo/sudo.log
$ touch /etc/sudoers.d/sudo_config
$ sudo nano /etc/sudoers.d/sudo_config
```

Add following line for authentication using sudo has to be limited to 3 attempts in the event of an incorrect password:

```
Defaults        passwd_tries=3
```

For wrong password warning message, add:

```
Defaults        badpass_message="Password is wrong, please try again!"
```

Each action log file has to be saved in the `/var/log/sudo/sudo.log` file:

```
Defaults        logfile="/var/log/sudo/sudo.log"
Defaults        log_input,log_output
Defaults        iolog_dir="/var/log/sudo"
```

Require tty: (*Why use tty? If some non-root code is exploited (a PHP script, for example), the `requiretty` option means that the exploit code won't be able to directly upgrade its privileges by running `sudo`.*)

```
Defaults        requiretty
```

For security reasons too, the paths that can be used by sudo must be restricted. Example :
`/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin` (It was already set there)

```
Defaults secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:
                        /usr/bin:/sbin:/bin:/snap/bin"
```

8. Change hostname

Check current hostname

```
$ hostnamectl
```

Change the hostname

```
$ hostnamectl set-hostname new_hostname
```

Change `/etc/hosts` file

```
$ sudo nano /etc/hosts
```


Change `old_hostname` with `new_hostname`:

```
127.0.0.1    localhost
127.0.0.1    new_hostname
```

Reboot and check the change

```
$ sudo reboot
```

9. The script monitoring.sh

 **Que es un script ?** Es una secuencia de comandos guardada en un fichero que cuando se ejecuta hará la función de cada comando.

9-1 Architecture

Para poder ver la arquitectura del SO y su versión de kernel utilizaremos el comando

```
uname -a
```

("a" == "--all") que básicamente muestra toda la información excepto si el tipo de procesador es desconocido o la plataforma de hardware.

9-2 Núcleos físicos

Para poder mostrar el número de núcleos físicos haremos uso del fichero /proc/cpuinfo el cual proporciona información acerca del procesador: su tipo, marca, modelo, rendimiento, etc. Usaremos el comando

```
grep "physical id" /proc/cpuinfo | wc -l
```

con el comando grep buscaremos dentro del fichero "physical id" y con wc -l contaremos las líneas del resultado de grep. Esto lo hacemos ya que la manera de cuantificar los núcleos no es muy común. Si hay un procesador marcará 0 y si tiene más de un procesador, mostrará toda la información del procesador por separado contando los procesadores usando la notación cero. De esta manera simplemente contaremos las líneas que hay ya que es más cómodo cuantificarlo así.

9-3 Núcleos virtuales

Para poder mostrar el número de núcleos virtuales es muy parecido al anterior. Haremos uso de nuevo del fichero /proc/cpuinfo , pero, en este caso utilizaremos el comando

```
grep "processor" /proc/cpuinfo | wc -l
```

El uso es prácticamente el mismo al anterior solo que en vez de contar las líneas de "physical id" lo haremos de "processor". Lo hacemos así por el mismo motivo de antes, la manera de cuantificar marca 0 si hay un procesador.

9-4 Memoria RAM

Para mostrar la memoria ram haremos uso del comando free para así ver al momento información sobre la ram, la parte usada, libre, reservada para otros recursos, etc. Para más info sobre el comando pondremos free --help. Nosotros daremos uso de free --mega (o -m) ya que en el subject aparece esa unidad de medida.

Una vez hemos ejecutado este comando debemos filtrar nuestra búsqueda ya que no necesitamos toda la información que nos aporta, lo primero que debemos mostrar es la memoria usada, para ello haremos uso del comando awk que lo que hace este comando es para procesar datos basados en archivos de texto, es decir, podremos utilizar los datos que nos interesen de X fichero. Por último, lo que haremos será comparar si la primera palabra de una fila es igual a "Mem:" mostraremos la tercera palabra de esa fila que será la memoria usada.

Todo el comando junto sería

```
free --mega | awk '$1 == "Mem:" {print $3}'
```

En el script el valor de retorno de este comando se lo asignaremos a una variable que concatenaremos con otras variables para que todo quede igual como especifica el subject.

Para obtener la memoria total el comando es prácticamente igual al anterior lo único que deberemos cambiar es que en vez de mostrar la tercera palabra de la fila queremos la segunda

```
free --mega | awk '$1 == "Mem:" {print $2}'
```

Por última parte debemos calcular el % de memoria usada. El comando de nuevo es parecido a los anteriores la única modificación que haremos en la parte de mostrar. Como la operación para conseguir el tanto por ciento no es exacta nos puede dar muchos decimales y en el subject solo aparecen 2 así que nosotros haremos lo mismo, por eso utilizamos `%.2f` para que así solo se muestren 2 decimales.. Todo el comando sería

```
free --mega | awk '$1 == "Mem:" {printf("%.2f", $3/$2*100)}'
```

9-5 Memoria del disco

Para poder ver la memoria del disco ocupada y disponible utilizaremos el comando `df` que significa "disk filesystem", se utiliza para obtener un resumen completo del uso del espacio en disco. Como en el subject indica la memoria utilizada se muestra en MB así que entonces utilizaremos el flag `-m`. Acto seguido haremos un `grep` para que solo nos muestre las líneas que contengan `"/dev/"` y seguidamente volveremos a hacer otro `grep` con el flag `-v` para excluir las líneas que contengan `"/boot"`. Por último, utilizaremos el comando `awk` y sumaremos el valor de la tercera palabra de cada línea para que, una vez sumadas todas las líneas, mostrar el resultado final de la suma. El comando entero es el siguiente:

```
df -m | grep "/dev/" | grep -v "/boot" |  
  
awk '{mem_use += $3} END {print mem_use}'
```

Para obtener el espacio total utilizaremos un comando muy parecido. Las únicas diferencias serán que los valores que sumaremos serán los `$2` en vez de `$3` y la otra diferencia es que en el subject aparece el tamaño total en Gb así que como el resultado de la suma nos da el numero en Mb debemos transformarlo a Gb, para ello debemos dividir el numero entre 1024 y quitar los decimales.

```
df -m | grep "/dev/" | grep -v "/boot" |  
  
awk '{mem_total += $2} END {printf("%.1f", mem_total/1024)}'
```

Por último, debemos mostrar un porcentaje de la memoria usada. Para ello, de nuevo, utilizaremos un comando muy parecido a los dos anteriores. Lo único que cambiaremos es que combinaremos los dos comandos anteriores para tener dos variables, una que representa la memoria usada y la otra la total. Hecho esto haremos una operación para conseguir el tanto por ciento `use/total*100` y el resultado de esta operación lo mostraremos como aparece en el subject, entre paréntesis y con el símbolo `%` al final. El comando final es este:

```
df -m | grep "/dev/" | grep -v "/boot" |  
  
awk '{use += $3} {total += $2} END {printf("%d", use/total*100)}'
```

9-6 Porcentaje uso de CPU

Para poder ver el porcentaje de uso de CPU haremos uso del comando `vmstat` que muestra estadísticas del sistema, permitiendo obtener un detalle general de los procesos, uso de memoria, actividad de CPU, estado del sistema, etc. También daremos uso del comando `tail -1` que nos va a permitir es que solo produzca el output de una línea. Por último, solo mostraremos la palabra 15 que es el uso de memoria disponible.

El resultado de este comando solo es una parte del resultado final ya que todavía hay que hacer alguna operación en el script para que quede bien. Lo que habría que hacer es a 100 restarle la cantidad que nos ha devuelto nuestro comando, el resultado de esa operación lo mostraremos con un decimal y ya estaría hecha la operación.

```
vmstat | tail -1 | awk '{printf("%.1f", 100-$15)}'
```

9-7 Último reinicio

Para ver la fecha y hora de nuestro último reinicio haremos uso del comando `who` con el flag `-b` ya que con ese flag nos mostrará por pantalla el tiempo del último arranque del sistema. Como ya nos ha pasado anteriormente nos muestra más información de la que deseamos asique filtraremos y solo mostraremos lo que nos interesa, para ello haremos uso del comando `awk` que mostrará por pantalla la tercera palabra de esa línea, un espacio y la cuarta palabra (si lo tenemos configurado en español, hay que mostrar la cuarta y quinta palabra). El comando entero sería el siguiente:

```
who -b | awk '{print $3 " " $4}'  
who -b | awk '{print $4 " " $5}'
```

9-8 Uso LVM

Para comprobar si LVM está activo o no haremos uso del comando `lsblk`, éste nos muestra información de todos los dispositivos de bloque (discos duros, SSD, memorias, etc.) entre toda la información que proporciona podemos ver `lvm` en el tipo de gestor. Para este comando haremos un `if` que mostrará Yes o No. Básicamente la condición que busquemos será contar el número de líneas en las que aparece "lvm" y si hay más de 0 mostramos Yes, si hay 0 se mostrará No. Todo el comando sería:

```
if [ $(lsblk | grep "lvm" | wc -l) -gt 0 ]; then echo yes; else echo no; fi
```

9-9 Conexiones TCP

Para mirar el número de conexiones TCP establecidas. Utilizaremos el comando `ss` sustituyendo al ya obsoleto `netstat`. Filtraremos con el flag `-ta` para que solo se muestren las conexiones TCP. Por último, haremos un `grep` para ver las que están establecidas ya que también hay solo de escucha y cerraremos con `wc -l` para que cuente el número de líneas. El comando queda tal que así:

```
ss -ta | grep ESTAB | wc -l
```


9-10 Número de usuarios

Daremos uso del comando `users` que nos mostrará el nombre de los usuarios que hay, sabiendo esto, pondremos `wc -w` para que cuente la cantidad de palabras que hay en la salida del comando. El comando entero queda así

```
users | wc -w
```

9-11 Dirección IP y MAC

Para obtener la dirección del host haremos uso del comando `hostname -I` y para obtener la MAC haremos uso del comando `ip link` que se utiliza para mostrar o modificar las interfaces de red. Como aparecen más de una interfaz, IP's etc. Utilizaremos el comando `grep` para buscar lo que deseamos y así poder mostrar por pantalla solo lo que nos piden. Para ello pondremos

```
ip link | grep "link/ether" | awk '{print $2}'
```

y de esta manera solo mostraremos la MAC.

9-12 Número de comandos ejecutados con sudo

Para poder obtener el número de comandos que son ejecutados con `sudo` haremos uso del comando `journalctl` que es una herramienta que se encarga de recopilar y administrar los registros del sistema. Acto seguido pondremos `_COMM=sudo` para así filtrar las entradas especificando su ruta. En nuestro ponemos `_COMM` ya que hace referencia a un script ejecutable. Una vez tengamos filtrada la búsqueda y solo aparezcan los registros de `sudo` todavía deberemos filtrar un poco más ya que cuando inicias o cierras sesión de root también aparece en el registro, entonces para terminar de filtrar pondremos un `grep COMMAND` y así solo aparecerán las líneas de comandos. Por último, pondremos `wc -l` para que así nos salgan enumeradas las líneas. El comando entero es el siguiente:

```
journalctl _COMM=sudo | grep COMMAND | wc -l)
```

Para comprobar que funcione correctamente podemos correr el comando en el terminal, poner un comando que incluya `sudo` y volver a correr el comando y deberá incrementar el número de ejecuciones de `sudo`.

```
root@gemartin42:/home/gemartin# journalctl _COMM=sudo | grep COMMAND | wc -l
36
root@gemartin42:/home/gemartin# sudo apt install
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@gemartin42:/home/gemartin# journalctl _COMM=sudo | grep COMMAND | wc -l
37
root@gemartin42:/home/gemartin# _
```

9-13 monitoring.sh

```
#!/bin/bash

# ARCH
arch=$(uname -a)

# CPU PHYSICAL
cpuf=$(grep "physical id" /proc/cpuinfo | wc -l)

# CPU VIRTUAL
cpuv=$(grep "processor" /proc/cpuinfo | wc -l)

# RAM
ram_total=$(free --mega | awk '$1 == "Mem:" {print $2}')
ram_use=$(free --mega | awk '$1 == "Mem:" {print $3}')
ram_percent=$(free --mega | awk '$1 == "Mem:" {printf("%.2f", $3/$2*100)}')

# DISK
disk_total=$(df -m | grep "/dev/" | grep -v "/boot" |
    awk '{disk_t += $2} END {printf("%.1f", disk_t/1024)}')
disk_use=$(df -m | grep "/dev/" | grep -v "/boot" |
    awk '{disk_u += $3} END {print disk_u}')
disk_percent=$(df -m | grep "/dev/" | grep -v "/boot" |
    awk '{disk_u += $3} {disk_t += $2} END {printf("%d", disk_u/disk_t*100)}')

# CPU LOAD
cpul=$(vmstat | tail -1 | awk '{printf("%.1f", 100-$15)}')

# LAST BOOT
lb=$(who -b | awk '$1 == "system" {print $3 " " $4}')

# LVM USE
lvmu=$(if [ $(lsblk | grep "lvm" | wc -l) -gt 0 ]; then echo yes; else echo no;
fi)

# TCP CONNEXIONS
tcpc=$(ss -ta | grep ESTAB | wc -l)


# USER LOG
ulog=$(users | wc -w)

# NETWORK
ip=$(hostname -I)
mac=$(ip link | grep "link/ether" | awk '{print $2}')

# SUDO
cmdnd=$(journalctl _COMM=sudo | grep COMMAND | wc -l)

wall    "
        #Architecture: $arch
        #CPU physical: $cpuf
        #vCPU: $cpuv
        #Memory usage: $ram_use/${ram_total} Mb ($ram_percent %)
        #Disk usage: $disk_use/${disk_total} Gb ($disk_percent %)
        #CPU load: $cpul %
        #Last boot: $lb
        #LVM use: $lvmu
        #Connections TCP: $tcpc ESTABLISHED
        #User log: $ulog
        #Network: IP $ip ($mac)
        #Sudo: $cmdnd cmd
    "
```

10. Crontab configuration

 **Que es crontab?** Es un administrador de procesos en segundo plano. Los procesos indicados serán ejecutados en el momento que especifiques en el fichero crontab.

1. Place monitoring.sh in /usr/local/bin/

2. Open crontab

```
$ sudo crontab -u root -e
```

3. Replace

```
23 # m h dom mon dow    command
```

with:

```
*/10 * * * * sh /usr/local/bin/monitoring.sh
```

4. Start / Stop

```
$sudo /etc/init.d/cron start
```

```
$sudo /etc/init.d/cron stop
```

Funcionamiento de cada parámetro de crontab:

* * * * * command

* ➤ indica en los minutos en los que se ejecutará. Si se deja en * se ejecutará cada minuto. Si se pone un número (por ejemplo 30) se ejecutará cada vez que en la hora completa haya un 30, es decir cada hora (en nuestro ejemplo, se ejecutará a las 17:30 y a las 18:30, pero no a las 18:00). Para indicar que se quiere ejecutar cada cierto periodo en minutos hay que utilizar */x donde x sería el periodo en minutos. Así, en born2beroot lo que hay que poner en el primer * es */10

* ➤ indica en qué horas queremos que se ejecute. En nuestro caso sería en todas, por lo que se queda *

* ➤ hace referencia al día del mes, por ejemplo, se puede especificar 15 si se quiere ejecutar cada día 15.

* ➤ indica en que día del mes queremos que se ejecute. En nuestro caso sigue siendo en todos por lo que se mantiene el *.

* ➤ indica qué mes se ejecutará. Lo mismo * porque es en todos.

* ➤ indica qué día de la semana. Lo mismo * porque es en todos

command ➤ Refiere al comando o a la ruta absoluta del script a ejecutar.

Check *root*'s scheduled *cron* jobs via:

```
$ sudo crontab -u root -l
```

11. How do you get the Machine signature?


Para obtener la firma lo primero que debemos hacer es apagar la máquina virtual ya que una vez la enciendas o modifies algo la firma cambiará.

El siguiente paso será ubicarnos en la ruta donde tengamos el archivo .vdi de nuestra máquina virtual.

```
gemartin@cbr12s2 gemartin % ls
Born2beroot.vbox      Born2beroot.vbox-prev  Born2beroot.vdi
gemartin@cbr12s2 gemartin % pwd
/sgoinfre/Person/gemartin
gemartin@cbr12s2 gemartin %
```

Por último, haremos `shasum nombremaquina.vdi` y esto nos dará la firma. El resultado de esta firma es lo que tendremos añadir a nuestro fichero `signature.txt` para posteriormente subir el fichero al repositorio de la intra.

Muy importante no volver a abrir la máquina ya que se modificará la firma. Para las correcciones recuerda clonar la máquina ya que así podrás encenderla sin miedo a que cambie la firma.

 **Que es shasum ?** Es un comando que permite identificar la integridad de un fichero mediante la suma de comprobación del hash SHA-1 de un archivo.

```
gemartin@cbr12s2 gemartin % shasum Born2beroot.vdi
28169292244a4498cff84716cdc5123c15f08c5b  Born2beroot.vdi
gemartin@cbr12s2 gemartin %
```

Para generar el archivo `signatura.txt` tendremos que redirigir la salida de `shasum` con el operador `>`. El comando quedaría así

```
$> shasum nombredelamaquina.vdi > signatura.txt
```

BONUS

Lighttpd

🧠 **Qué es Lighttpd ?** Es un servidor web diseñado para ser rápido, seguro, flexible, y fiel a los estándares. Está optimizado para entornos donde la velocidad es muy importante. Esto se debe a que consume menos CPU y memoria RAM que otros servidores.

1 ° Instalación de paquetes de lighttpd.

```
sudo apt install lighttpd
```

2 ° Permitimos las conexiones mediante el puerto 80 con el comando

```
sudo ufw allow 80.
```

3 ° Comprobamos que realmente hayamos permitido. Debe aparecer el puerto 80 y allow.

4 ° Añadimos a la máquina la regla que incluya el puerto 80.

Nombre	Protocolo	IP anfitrión	Puerto anfitrión	IP invitado	Puerto invitado
Rule 1	TCP		4242		4242
Rule 2	TCP		80		80

Mariadb

🧠 **Qué es MariaDB ?** Es una base de datos. Se utiliza para diversos fines, como el almacenamiento de datos, el comercio electrónico, funciones a nivel empresarial y las aplicaciones de registro.

Now it's time to install a database manager. First we install it with the following command:

```
sudo apt install mariadb-server
```

Then, once the installation is complete, the `mysql_secure_installation` script must be run to define a new key for the root user and other configurations.

```
sudo mysql_secure_installation
```


After defining the new password for the root user, you will have to answer a few questions.

```
Remove anonymous users? Y
Disallow root login remotely? Y
Remove test database and access to it? Y
Reload privilege tables now? Y
```

Now access the MariaDB console to create the database for WordPress and a new user.

```
sudo mysql -u root -p
> CREATE DATABASE wordpress;
> GRANT ALL PRIVILEGES on wordpress.* TO 'wp_user'@'localhost' IDENTIFIED BY
'wordpress_pss123';
> FLUSH PRIVILEGES;
> EXIT;
```

PHP

 **Qué es PHP ?** PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML.

1. Instalación de los paquetes necesarios

En primer lugar se instalarán todos los paquetes necesarios, aunque antes de eso se deben actualizar los repositorios, para instalar las últimas versiones. Actualizarlos es tan simple como ejecutar en una terminal:

```
sudo apt update
```

Una vez actualizados los repositorios, procederemos a la instalación en sí. Necesitaremos los paquetes correspondientes a, PHP y MySQL. Con este comando los instalaremos todos:

```
sudo apt install mysql-server php-common php-cgi php php-mysql
```

Los paquetes que se instalarán con ese comando son:

- `mysql-server` → servidor de base de datos
- `php-common`; `php` → paquetes para PHP, un lenguaje de programación server-side
- `php-cgi` → Permite usar PHP desde Lighttpd
- `php-mysql` → Permite usar MySQL desde Lighttpd

Una vez el comando termine de ejecutarse, todos los paquetes estarán instalados y configurados; preparados para ser usados.

2. Configuraciones

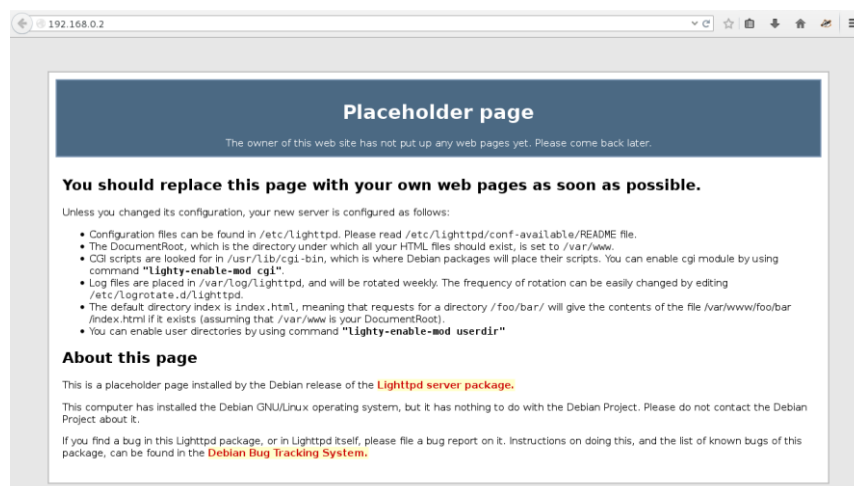
Ahora con los paquetes ya listos, procederemos a habilitar PHP en Lighttpd (gracias al paquete `php-cgi`). Hacerlo es así de simple:

```
sudo lighty-enable-mod fastcgi-php
```

Tras esto, reiniciaremos Lighttpd para que los cambios surtan efecto:

```
sudo service lighttpd reload
```

Esto nos deja con un servidor web ya funcional, el cual podemos probar entrando a la dirección IP de la máquina o dispositivo en el que hayamos instalado el Lighttpd. Puede verse lo siguiente:



2.1 Configuración de permisos

Es útil (y necesario para posteriores pasos) cambiar los permisos del directorio del servidor (por defecto /var/www) para poder modificar archivos y carpetas sin necesidad de ser root.

Para ello tendremos que ejecutar los siguientes comandos:

```
sudo chown www-data:www-data /var/www
sudo chmod 775 /var/www
```

El primer comando cambiará el propietario y el grupo de /var/www a www-data (el usuario que utiliza el servidor web). El segundo comando permitirá a cualquier usuario perteneciente al grupo www-data modificar el directorio.

2.2 Configuraciones específicas de Wordpress

Antes de instalar Wordpress, es necesario modificar unas pocas cosas para asegurar su funcionamiento. En primer lugar, modificaremos el archivo de configuración principal de Lighttpd, situado en /etc/lighttpd/lighttpd.conf. Allí nos encontraremos, entre otras cosas, lo siguiente:

```
server.modules = (
    "mod_access",
    "mod_alias",
    "mod_compress",
    "mod_redirect",
    # "mod_rewrite",
)
```


Lo único que tenemos que hacer aquí es cambiar la línea de

```
# "mod_rewrite",
```

Por

```
"mod_rewrite",
```

Wordpress

 **Qué es Wordpress ?** Es un sistema de gestión de contenidos enfocado a la creación de cualquier tipo de página web.

Escribe estos comandos:

```
cd /var/www/html

sudo wget https://wordpress.org/latest.tar.gz

sudo tar -xf latest.tar.gz

sudo chown -R www-data:www-data /var/www/html/wordpress/

sudo chmod 755 -R /var/www/html/wordpress/
```

First, let's rename the WordPress folder with the name of the virtual host you want. In this case I will choose **wordpress.osradar.lan**. Obviously, you are free to choose yours.

```
sudo mv /var/www/html/wordpress/ /var/www/html/wordpress.osradar.lan
```

Then, create the folder dedicated to virtual hosts in Lighttpd.

```
sudo mkdir -p /etc/lighttpd/vhosts.d/
```

And add it to the main Lighttpd configuration file:

```
sudo nano /etc/lighttpd/lighttpd.conf
include_shell "cat /etc/lighttpd/vhosts.d/*.conf"
```

Save the changes and close the file.

Now create the configuration file for the new Virtual host.

```
sudo nano /etc/lighttpd/vhosts.d/wordpress.osradar.lan.conf
```

Note how the file name equals the WordPress folder. This is for recommendation. Now add the following:

```
$HTTP["host"] =~ "^(|.)*wordpress.osradar.lan$" {
server.document-root = "/var/www/html/wordpress.osradar.lan"
server.errorlog = "/var/log/lighttpd/wordpress.osradar.lan-error.log"
url.rewrite-final = ("^/(.*.php)" => "$0", "^/(.*)$" => "/index.php/$1" )
}
```

2.- Creating a new virtual host for WordPress Save the changes and close the file.

Now, check the syntax.

```
sudo lighttpd -t -f /etc/lighttpd/lighttpd.conf
Syntax OK
```

And restart Lighttpd.

```
sudo systemctl restart lighttpd
sudo adduser ftp ftpgroup
```


HOJA DE CORRECCIÓN

Preliminaries

If cheating is suspected, the evaluation stops here. Use the "Cheat" flag to report it. Take this decision calmly, wisely, and please, use this button with caution.

Preliminary tests

- Defense can only happen if the student being evaluated or group is present. This way everybody learns by sharing knowledge with each other.
- If no work has been submitted (or wrong files, wrong directory, or wrong filenames), the grade is 0, and the evaluation process ends.
- For this project, you have to clone their Git repository on their station.

 Yes

 No

General instructions

General instructions

- During the defense, as soon as you need help to verify a point, the student evaluated must help you.
- Ensure that the "signature.txt" file is present at the root of the cloned repository.
- Check that the signature contained in "signature.txt" is identical to that of the ".vdi" file of the virtual machine to be evaluated. A simple "diff" should allow you to compare the two signatures. If necessary, ask the student being evaluated where their ".vdi" file is located.
- As a precaution, you can duplicate the initial virtual machine in order to keep a copy.
- Start the virtual machine to be evaluated.
- If something doesn't work as expected or the two signatures differ, the evaluation stops here.

 Yes

 No

Mandatory part

The project consists of creating and configuring a virtual machine following strict rules. The student being evaluated will have to help you during the defense. Make sure that all of the following points are observed.

Project overview

- The student being evaluated should explain to you simply:
 - How a virtual machine works.
 - Their choice of operating system.
 - The basic differences between CentOS and Debian.
 - The purpose of virtual machines.
 - If the evaluated student chose CentOS: what SELinux and DNF are.
 - If the evaluated student chose Debian: the difference between aptitude and apt, and what APPArmor is. During the defense, a script must display information all every 10 minutes. Its operation will be checked in detail later. If the explanations are not clear, the evaluation stops here.

 Yes

 No

Simple setup

Remember: Whenever you need help checking something, the student being evaluated should be able to help you.

- Ensure that the machine does not have a graphical environment at launch. A password will be requested before attempting to connect to this machine. Finally, connect with a user with the help of the student being evaluated. This user must not be root. Pay attention to the password chosen, it must follow the rules imposed in the subject.
- Check that the UFW service is started with the help of the evaluator.
- Check that the SSH service is started with the help of the evaluator.
- Check that the chosen operating system is Debian or CentOS with the help of the evaluator. If something does not work as expected or is not clearly explained, the evaluation stops here.



Yes



No

User

Remember: Whenever you need help checking something, the student being evaluated should be able to help you.

The subject requests that a user with the login of the student being evaluated is present on the virtual machine. Check that it has been added and that it belongs to the "sudo" and "user42" groups.

Make sure the rules imposed in the subject concerning the password policy have been put in place by following the following steps.

First, create a new user. Assign it a password of your choice, respecting the subject rules. The student being evaluated must now explain to you how they were able to set up the rules requested in the subject on their virtual machine.

Normally there should be one or two modified files. If there is any problem, the evaluation stops here.

- Now that you have a new user, ask the student being evaluated to create a group named "evaluating" in front of you and assign it to this user. Finally, check that this user belongs to the "evaluating" group.
- Finally, ask the student being evaluated to explain the advantages of this password policy, as well as the advantages and disadvantages of its implementation. Of course, answering that it is because the subject asks for it does not count.

If something does not work as expected or is not clearly explained, the evaluation stops here.



Yes



No

Hostname and partitions

Remember: Whenever you need help checking something, the student being evaluated should be able to help you.

- Check that the hostname of the machine is correctly formatted as follows: login42 (login of the student being evaluated).
- Modify this hostname by replacing the login with yours, then restart the machine. If on restart, the hostname has not been updated, the evaluation stops here.
- You can now restore the machine to the original hostname.
- Ask the student being evaluated how to view the partitions for this virtual machine.
- Compare the output with the example given in the subject. Please note: if the student evaluated makes the bonuses, it will be necessary to refer to the bonus example.

This part is an opportunity to discuss the scores! The student being evaluated should give you a brief explanation of how LVM works and what it is all about.

If something does not work as expected or is not clearly explained, the evaluation stops here.



Yes



No

SUDO

Remember: Whenever you need help checking something, the student being evaluated should be able to help you.

- Check that the "sudo" program is properly installed on the virtual machine.
- The student being evaluated should now show assigning your new user to the "sudo" group.
- The subject imposes strict rules for sudo. The student being evaluated must first explain the value and operation of sudo using examples of their choice. In a second step, it must show you the implementation of the rules imposed by the subject.
- Verify that the "/var/log/sudo/" folder exists and has at least one file. Check the contents of the files in this folder. You should see a history of the commands used with sudo. Finally, try to run a command via sudo. See if the file (s) in the "/var/log/sudo/" folder have been updated. If something does not work as expected or is not clearly explained, the evaluation stops here.

✓ Yes

✗ No

UFW

Remember: Whenever you need help checking something, the student being evaluated should be able to help you.

- Check that the "UFW" program is properly installed on the virtual machine.
- Check that it is working properly.
- The student being evaluated should explain to you basically what UFW is and the value of using it.
- List the active rules in UFW. A rule must exist for port 4242.
- Add a new rule to open port 8080. Check that this one has been added by listing the active rules.
- Finally, delete this new rule with the help of the student being evaluated. If something does not work as expected or is not clearly explained, the evaluation stops here.

✓ Yes

✗ No

SSH

Remember: Whenever you need help checking something, the student being evaluated should be able to help you.

- Check that the SSH service is properly installed on the virtual machine.
- Check that it is working properly.
- The student being evaluated must be able to explain to you basically what SSH is and the value of using it.
- Verify that the SSH service only uses port 4242.
- The student being evaluated should help you use SSH in order to log in with the newly created user. To do this, you can use a key or a simple password. It will depend on the student being evaluated. Of course, you have to make sure that you cannot use SSH with the "root" user as stated in the subject. If something does not work as expected or is not clearly explained, the evaluation stops here.

✓ Yes

✗ No

Script monitoring

Remember: Whenever you need help checking something, the student being evaluated should be able to help you.

The student being evaluated should explain to you simply:

- How their script works by showing you the code.
- What "cron" is.
- How the student being evaluated set up their script so that it runs every 10 minutes from when the server starts. Once the correct functioning of the script has been verified, the student being evaluated should ensure that this script runs every minute. You can run whatever you want to make sure the script runs with dynamic values correctly. Finally, the student being evaluated should make the script stop running when the server has started up, but without modifying the script itself. To check this point, you will have to restart the server one last time. At startup, it will be necessary to check that the script still exists in the same place, that its rights have remained unchanged, and that it has not been modified. If something does not work as expected or is not clearly explained, the evaluation stops here.

✓ Yes

✗ No

Bonus

Evaluate the bonus part if, and only if, the mandatory part has been entirely and perfectly done, and the error management handles unexpected or bad usage. In case all the mandatory points were not passed during the defense, bonus points must be totally ignored.

Bonus

Check, with the help of the subject and the student being evaluated, the bonus points authorized for this project:

- Setting up partitions is worth 2 points.
- Setting up WordPress, only with the services required by the subject, is worth 2 points.
- The free choice service is worth 1 point. Verify and test the proper functioning and implementation of each extra service. For the free choice service, the student being evaluated has to give you a simple explanation about how it works and why they think it is useful. Please note that NGINX and Apache2 are prohibited.

Rate it from 0 (failed) through 5 (excellent)

0

RESPUESTAS DE LA EVALUACIÓN

▪ Qué es una máquina virtual ?

Es un software que simula un sistema de computación y puede ejecutar programas como si fuese una computadora real. Permite crear múltiples entornos simulados o recursos dedicados desde un solo sistema de hardware físico. Su objetivo es el de proporcionar un entorno de ejecución independiente de la plataforma de hardware y del sistema operativo, que oculte los detalles de la plataforma subyacente y permita que un programa se ejecute siempre de la misma forma sobre cualquier plataforma.

▪ Por qué has escogido Debian ?

Esto es algo personal para cada uno, mi opinión: El propio subject explica que es más sencillo hacerlo en Debian y si buscas documentación/tutoriales hay muchos y todos se han hecho en Debian.

▪ Diferencias básicas entre CentOS y Debian

CentOs	Debian
Más estable. Respaldado por la comunidad Red Hat	Repaldado por la comunidad libre
Es una bifurcación de la destribución de Red Hat. Red Hat saca su propia distribución pero luego abre el código. Entonces unos voluntarios cogen ese código, quitan las referencias a Red Hat y sacan distribución gratuita. Por lo que CentOs es totalmente compatible con la distribución de RedHat	Se inició en 1993 y es un sistema operativo de código abierto totalmente. Desarrollado por voluntarios.
Por lo tanto tiene una distribución mayor.	Al ser código abierto tiene menor distribución. Es como comparar el paquete office de Microsoft con LibreOffice
Actualmente es más seguro y es lo que se usa en servidores	La comunidad de código abierto está trabajando bastante y ya se en
El formato de paquetes son RPM	El formato de paquetes es DEB
El formato de disco por defecto es XFS	El formato de disco por defecto es EXT4
Al estar basado en la distribución de Red Hat no hay actualizaciones parciales. Esto lo hace menos compatible con software moderno ya que los ciclos de actualización son largos, aprox 5 años	Salen más distribuciones y los bugs se van corrigiendo por la comunidad. Además, se pueden sacar actualizaciones para hacerlo compatible con nuevas aplicaciones
Difícil de actualizar. Se recomienda reinstalación completa de la nueva versión.	Fácil de actualizar
	Mejor interfaz gráfico

Conclusión - CentOS vs Debian

Este artículo de CentOS vs Debian cubre casi todas las razones específicas del negocio o basadas en la elección para diferenciar entre CentOS y Debian. Tanto Debian vs CentOS es una excelente pieza de software y es utilizada por miles de aplicaciones y muchos más desarrolladores. Tanto Debian vs CentOS son altamente confiables por la industria y ejecutan los componentes centrales de las aplicaciones de misión crítica. Por lo tanto, no importa mucho cuál se usa en un escenario dado. Un desarrollador puede preferir con qué se siente más cómodo y sobre cuál sabe más. Si se proporciona una pauta general, CentOS probablemente ejecuta una cantidad de servidores actualmente que cualquier otra versión de Linux. También para un principiante, aprender y comenzar con CentOS tiene más sentido y proporciona un alcance profesional mejor y más desafiante. Ubuntu es un importante punto de brownie para Debian. Desde la perspectiva del administrador también, CentOS gana en la mayoría de las situaciones en comparación con Debian.

▪ **Diferencias entre apt y aptitude**

APT viene de Advanced Package Tool. Que es el gestor de paquetes de Debian y que se utilizaba con el comando apt-get. apt-get derivó a apt con algunas funciones mejoradas, pero sigue siendo a bajo nivel

Aptitude está basado en apt es más de alto nivel y hace otras funciones automáticamente, como actualizar la lista de paquetes, marca paquetes como instalado, no instalado. Indica dependencias entre paquetes...

▪ **Qué es APPArmor ?**

Es un módulo de seguridad del kernel Linux que permite al administrador del sistema restringir las capacidades de un programa.

▪ **Qué es LVM ?**

Es un gestor de volúmenes lógicos. Proporciona un método para asignar espacio en dispositivos de almacenamiento masivo, que es más flexible que los esquemas de particionado convencionales para almacenar volúmenes.

▪ **Diferencias entre SELinux y AppArmor**

Ambos son módulos de seguridad para controlar el acceso que tienen los programas a archivos.

AppArmor surgió como alternativa de código cerrado a SELinux debido a la complejidad del segundo. Posteriormente el código fue liberado y actualmente lo mantiene Canonical

El funcionamiento de Selinux se basa en añadir etiquetas a los archivos mientras que AppArmor se basa en control por ruta.

AppArmor es más fácil de utilizar. Sólo se recomienda el uso de Selinux a administradores de sistemas avanzados.

▪ Comandos

1 ° Comprobar que no haya ninguna interfaz gráfica en uso.

Utilizaremos el siguiente comando y nos debe aparecer el resultado “/usr/bin/dbus-run-session”. Si aparece algo diferente se está utilizando una interfaz gráfica.

```
ls /usr/bin/*session
```

2 ° Comprobar que el servicio UFW está en uso.

```
sudo service ufw status
```

3 ° Comprobar que el servicio SSH está en uso.

```
sudo service ssh status
```

4 ° Comprobar que utilizas el sistema operativo Debian o CentOS.

```
uname -v
```

5 ° Comprobar que tu usuario este dentro de los grupos "sudo" y "user42".

```
getent group <nombre del grupo>
```

6 ° Crear un nuevo usuario y mostrar que sigue la política de contraseñas que hemos creado.

```
sudo adduser <name_user> e introducimos una contraseña que siga la política.
```

7 ° Creamos un nuevo grupo llamado "evaluating".

```
sudo addgroup evaluating
```

8 ° Añadimos el nuevo usuario al nuevo grupo.

```
sudo adduser <name_user> evaluating
```

Para comprobar que se haya introducido correctamente.

```
sudo getent group evaluating
```

9 ° Comprobar que el hostname de la máquina es correcto login42.

```
sudo hostname
```

10 ° Modificar hostname para reemplazar tu login por el del evaluador.

```
sudo nano /etc/hostname y reemplazamos nuestro login por el nuevo.
```

```
sudo nano /etc/hosts y reemplazamos nuestro login por el nuevo.
```

Reiniciamos la máquina.

```
sudo reboot
```

Una vez nos hemos logueado de nuevo podemos ver cómo el hostname se ha cambiado correctamente.

11 ° Comprobar que todas las particiones son como indica el subject.

```
lsblk
```

12 ° Comprobar que sudo está instalado.

```
which sudo
```

Utilizar which realmente no es una buena práctica ya que no todos los paquetes se encuentran en las rutas donde which busca, aun así, para la evaluación es mejor ya que es un comando sencillo y fácil de aprender. Para un mejor uso haremos uso del siguiente comando:

```
dpkg -s sudo
```

13 ° Introducimos el nuevo usuario dentro del grupo sudo.

```
sudo adduser <name_user> sudo
```

Comprobamos que está dentro del grupo.

```
sudo getent group sudo
```

14 ° Muestra la aplicación de las reglas impuestas para sudo por el subject.

```
sudo nano /etc/sudoers.d/sudo_log
```

15 ° Muestra que la ruta /var/log/sudo/ existe y contiene al menos un fichero, en este se debería ver un historial de los comandos utilizados con sudo.

```
cd /var/log/sudo
```

```
cat sudo_log
```

Ejecuta un comando con sudo y comprueba que se actualiza el fichero.

```
cat sudo_log
```

16 ° Comprueba que el programa UFW está instalado en la máquina virtual y comprueba que funciona correctamente.

```
dpkg -s ufw
```

```
sudo service ufw status
```


17 ° Lista las reglas activas en UFW si no está hecha la parte bonus solo debe aparecer la regla para el puerto 4242.

```
sudo service ufw status numbered
```

18 ° Crea una nueva regla para el puerto 8080. Comprueba que se ha añadido a las reglas activas y acto seguido puedes borrarla.

```
sudo ufw allow 8080 (para crearla)

sudo service ufw status numbered
```

Para borrar la regla debemos utilizar el comando

```
sudo ufw delete <num_rule>
```

Comprobamos que se ha eliminado y vemos el número de la siguiente regla que hay que borrar. Borramos de nuevo la regla. Comprobamos que sólo nos quedan las reglas requeridas en el subject.

19 ° Comprueba que el servicio ssh está instalado en la máquina virtual, que funciona correctamente y que solo funciona por el puerto 4242.

```
which ssh

sudo service ssh status
```

20 ° Usa ssh para iniciar sesión con el usuario recién creado. Asegúrate de que no puede usar ssh con el usuario root. Intentamos conectarnos por ssh con el usuario root pero no tenemos permisos.

Nos conectamos por ssh con el nuevo usuario con el comando

```
ssh <newuser>@localhost -p 4242
```

21 ° Modifica el tiempo de ejecución del script de 10 minutos a 1.

Ejecutamos el siguiente comando para así modificar el fichero crontab

```
sudo crontab -u root -e
```

Modificamos el primer parámetro, en vez de 10 lo cambiamos a 1.

22 ° Finalmente haz que el script deje de ejecutarse cuando el servidor se haya iniciado, pero sin modificar el script.

```
sudo /etc/init.d/cron stop
```

Si queremos que vuelva a ejecutarse:

```
sudo /etc/init.d/cron start
```