


Script

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

1- **Arquitectura:** `uname -a`.

Para ver la arquitectura del SO. Printa toda la info excepto si es desconocida.

2- **Núcleos Físicos:** `grep "physical id" /proc/cpuinfo | wc -l`

Mostrar el número de Núcleos físicos.

`grep "physical id"` → Buscaremos dentro del fichero la palabra physical id.

`/proc/cpuinfo` → Donde esta la info del procesador.

`wc -l` → Para contar las líneas.

3- **Núcleos virtuales:** `grep processor /proc/cpuinfo | wc -l`

Mostrar el número de Núcleos Virtuales.

`grep processor` → Buscaremos dentro del fichero la palabra processor.

`/proc/cpuinfo` → Donde esta la info del procesador.

`wc -l` → Para contar las líneas.

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

Mostrar la Memoria RAM, la parte usada, libre.

`Free --mega` → Para ver la información sobre la RAM, y que salga en megas.

`awk` → Gracias a este comando podemos usar los datos que necesitemos del fichero.

`'$1 == "Mem:" {print $3}'` → Compara la primera fila del fichero con MEM, si está ahí printará la tercera palabra.

Para la memoria TOTAL es igual, pero se printará el 2.

`free --mega | awk '$1 == "Mem:" {printf("%.2f%%)\n", $3/$2*100}{'` → Es para calcular el % de memoria usada. Para que nos muestran 2 decimales, usamos .2f% y le añadimos otro % para que este salga. Obtener el porcentaje de memoria usada.

- \$2 es el valor total de memoria.
- \$3 es el valor usado de la memoria.

Entonces, $\$3/\2 calcula el porcentaje de memoria usado.

5- **Memoria del Disco:** `df -m | grep "/dev/" | grep -v "/boot" | awk '{memory_use += $3} END {print memory_use}'`

Memoria del Disco Ocupada y libre.

df -m → Disk File System, se usa para obtener un resumen completo del uso del espacio en disco, y -m para que se muestre en megas.

grep "/dev/" → Usamos grep para que solo nos muestre las líneas que tengan dev.

grep -v "/boot" → Se usa para excluir las líneas que tengan "/boot".

awk '{memory_use += \$3} END {print memory_use}' → Usaremos awk para poder usar los datos y sumaremos el valor de la tercera palabra de cada línea para una vez sumados printa el resultado final.

Para obtener el espacio total: `df -m | grep "/dev/" | grep -v "/boot" | awk '{memory_result += $2} END {print ("%0fGb\n"), memory_result/1024}'`

→ Esta vez, sumaremos los valores 2 y el tamaño aparece en GB, y por eso dividimos entre 1024 y le quitamos los decimales.

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

Lo que cambiaremos esta vez, es que combinaremos los 2 comandos anteriores y así tendremos memoria usada y total. Y calcularemos el %.

6- **Porcentaje Uso de CPU:** `vmstat 1 4 | tail -1 | awk '{print $15}'`

Muestra el % de uso del CPU.

vmstat 1 3 → Muestra las estadísticas del sistema, obteniendo detalles de los procesos, uso de memoria, actividad CPU, estado del sistema etc. Y pondremos el intervalo de segundos de 1 a 3.

tail -1 → Este comando nos va a permitir es que solo produzca el output la última línea, es decir, de las 3 generadas solo se printará la última.

awk '{print \$15}' → Con awk podremos acceder a los datos donde imprimimos la palabra 15 que es el uso de memoria disponible.

7- **Último reinicio:** `who -b | awk '$1 == "system" {print $3 " " $4}'`

Para mostrar la fecha y hora de nuestro último reinicio.

who -b → Nos mostrará fecha y hora, y con la flag nos mostrará el tiempo del último arranque.

awk '\$1 == "system" {print \$3 " " \$4}' → Con el comando awk compararemos la primera palabra de una línea con system. Se printará la tercera palabra de esa línea, un espacio y la cuarta palabra.

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

Para comprobar que LVM (gestor de volumen lógico) está activo.

lsblk → Nos muestra información de todos los dispositivos de bloque (discos duros, SSD, memorias, etc.) en la información que nos dan, podemos ver lvm en el tipo de gestor.

Usamos el if, ya que printaremos Yes o No. La condición será contar el número de líneas en las que aparece lvm, y si hay más de 0 printamos Yes, si hay 0 printamos No.

9- **Conexiones TCP:** `ss -ta | grep ESTAB | wc -l`

Para mirar el número de conexiones TCP establecidas.

ss -ta → Utilizaremos el comando ss sustituyendo al ya obsoleto netstat.

Proporciona información sobre las conexiones de red, enrutamiento, estadísticas de interfaz, etc. `// -t` → solo conexiones tcp. `// -a` → Mostrar todas las conexiones, tanto activas como pasivas.

grep ESTAB | wc -l → Haremos un grep para ver las que están establecidas, ya que también hay solo de escucha. Y usaremos `ec -l` para contar las líneas.

10- **Número de Usuarios:** `users | wc -w`

Nos mostrará el nombre de los usuarios que hay, y `wc -w` para que cuente la cantidad de palabras.

11- **Dirección IP y MAC:** `ip link | grep "link/ether" | awk '{print $2}'`

Obtener la dirección IP y MAC.

Para mirar el host usaremos: `hostname -I | grep "link/ether" | awk '{print $2}'`

Para MAC e IP: ip link → Para mostrar o modificar las interfaces de red.

grep "link/ether" → Donde tiene que buscar.

awk '{print \$2}' → Printa solo lo que necesitamos.

12- **Número de comandos ejecutados con Sudo:**

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

journalctl → Encarga de recopilar y administrar los registros del sistema.

_COMM=sudo → Para filtrar las entradas especificando su ruta. Nosotros ponemos `_COMM` ya que hace referencia a un script ejecutable.

grep COMMAND → Para filtrar aún más, usaremos grep para que colonos salgan las líneas con comandos.

wc -l → Para que nos salgan enumeradas las líneas.

```

#!/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 += $2} END {printf ("%1fGb\n"), disk_t/1024}')
disk_use=$(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 1 2 | tail -1 | awk '{printf $15}')
cpu_op=$(expr 100 - $cpul)
cpu_fin=$(printf "%.1f" $cpu_op)

# 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
cmnd=$(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} ($disk_percent%)
        CPU load: $cpu_fin%
        Last boot: $lb
        LVM use: $lvmu
        Connections TCP: $tcpc ESTABLISHED
        User log: $ulog
        Network: IP $ip ($mac)
        Sudo: $cmnd cmd"

```

Crontab

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

Para tener correctamente crontab configurado debemos editar el fichero crontab con el siguiente comando `sudo crontab -u root -e`.

En el fichero debemos añadir el siguiente comando para que el script se ejecute cada 10 minutos `*10 * * * * sh`/ ruta del script.

m ➤ Corresponde al minuto en que se va a ejecutar el script, el valor va de 0 a 59.

h ➤ La hora exacta, se maneja el formato de 24 horas, los valores van de 0 a 23, siendo 0 las 12:00 de la medianoche. dom ➤ hace referencia al día del mes, por ejemplo se puede especificar 15 si se quiere ejecutar cada día 15.

dow ➤ Significa el día de la semana, puede ser numérico (0 a 7, donde 0 y 7 son domingo) o las 3 primeras letras del día en inglés: mon, tue, wed, thu, fri, sat, sun.


user ➤ Define el usuario que va a ejecutar el comando, puede ser root, u otro usuario diferente siempre y cuando tenga permisos de ejecución del script.

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

Signature

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

Por último haremos `shasum nombremáquina.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 modificara la firma. Para las correcciones recuerda clonar la máquina ya que así podrás encenderla sin miedo a que cambie la firma.

 **Qué 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.

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.

▪ Porque 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.

Cuál es el propósito de las máquinas virtuales ?

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.

▪ Diferencias entre apt y aptitude

Aptitude es una versión mejorada de apt. APT es un administrador de paquetes de nivel inferior y aptitude es un administrador de paquetes de alto nivel. Otra gran diferencia es la funcionalidad que ofrecen ambas herramientas. Aptitude ofrece una mejor funcionalidad en comparación con apt-get. Ambos son capaces de proporcionar los medios necesarios para realizar la gestión de paquetes. Sin embargo, si se busca un enfoque con más características, debería ser, Aptitude.

▪ 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.

COMANDOS

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

Utilizaremos el comando `ls /usr/bin/*session` y nos debe aparecer el mismo resultado que en la captura. Si aparece algo diferente se está utilizando una interfaz gráfica.

```
gemartin@gemartin42:~$ ls /usr/bin/*session
/usr/bin/dbus-run-session
gemartin@gemartin42:~$
```

2 ° Comprobar que el servicio UFW está en uso.

`sudo ufw status`

```
root@gemartin42:/home/gemartin# sudo ufw status
Status: active

To Action From
--
4242 ALLOW Anywhere
80 ALLOW Anywhere
4242 (v6) ALLOW Anywhere (v6)
80 (v6) ALLOW Anywhere (v6)

root@gemartin42:/home/gemartin# _
```

`sudo service ufw status`

```
root@gemartin42:/home/gemartin# sudo service ufw status
• ufw.service - Uncomplicated firewall
   Loaded: loaded (/lib/systemd/system/ufw.service; enabled; vendor preset: enabled)
   Active: active (exited) since Thu 2022-11-24 01:19:28 CET; 5min ago
     Docs: man:ufw(8)
   Process: 316 ExecStart=/lib/ufw/ufw-init start quiet (code=exited, status=0/SUCCESS)
  Main PID: 316 (code=exited, status=0/SUCCESS)
    CPU: 41ms

Nov 24 01:19:28 gemartin42 systemd[1]: Finished Uncomplicated firewall.
Warning: journal has been rotated since unit was started, output may be incomplete.
root@gemartin42:/home/gemartin# _
```

3 ◦ Comprobar que el servicio SSH está en uso.

`sudo service ssh status`

```
root@gemartin42:/home/gemartin# sudo service ssh status
• ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2022-11-24 01:19:30 CET; 7min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
   Process: 552 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 613 (sshd)
     Tasks: 1 (limit: 1127)
    Memory: 3.8M
       CPU: 18ms
    CGroup: /system.slice/ssh.service
           └─613 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

Nov 24 01:19:30 gemartin42 systemd[1]: Starting OpenBSD Secure Shell server...
Nov 24 01:19:30 gemartin42 sshd[613]: Server listening on 0.0.0.0 port 4242.
Nov 24 01:19:30 gemartin42 sshd[613]: Server listening on :: port 4242.
Nov 24 01:19:30 gemartin42 systemd[1]: Started OpenBSD Secure Shell server.
root@gemartin42:/home/gemartin#
```

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

`uname -v` o `uname --kernel-version`

```
root@gemartin42:~# uname -v
#1 SMP Debian 5.10.149-2 (2022-10-21)
root@gemartin42:~#
```

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

`getent group sudo`

`getent group user42`

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.

```
root@gemartin42:~# sudo adduser newuser
Adding user `newuser' ...
Adding new group `newuser' (1002) ...
Adding new user `newuser' (1001) with group `newuser' ...
Creating home directory `/home/newuser' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for newuser
Enter the new value, or press ENTER for the default
  Full Name []: _
```


7 ◦ Creamos un nuevo grupo llamado "evaluating".

`sudo addgroup evaluating`

```
root@gemartin42:~# sudo addgroup evaluating
Adding group `evaluating' (GID 1003) ...
Done.
root@gemartin42:~# _
```

8 ◦ Añadimos el nuevo usuario al nuevo grupo.

`sudo adduser name_user evaluating`

```
root@gemartin42:~# sudo adduser newuser evaluating
Adding user `newuser' to group `evaluating' ...
Adding user newuser to group evaluating
Done.
root@gemartin42:~# _
```

Para comprobar que se haya introducido correctamente.

```
root@gemartin42:~# getent group evaluating
evaluating:x:1003:newuser
root@gemartin42:~#
```

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

```
root@gemartin42:~# hostname
gemartin42
root@gemartin42:~#
```

10 ◦ Modificar hostname para remplazar tu login por el del evaluador. En este caso lo reemplazaré por student42.

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

```
root@gemartin42:/home/gemartin# sudo nano /etc/hostname
```

```
GNU nano 5.4 /etc/hostname *
student42
```

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

```
root@gemartin42:/home/gemartin# sudo nano /etc/hosts
```

```
GNU nano 5.4 /etc/hosts *
127.0.0.1    localhost
127.0.1.1    student42

# The following lines are desirable for IPv6 capable hosts
::1          localhost ip6-localhost ip6-loopback
ff02::1      ip6-allnodes
ff02::2      ip6-allrouters
```

Reiniciamos la máquina.

```
root@gemartin42:/home/gemartin# sudo reboot_
```

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

```
gemartin@student42:~$ hostname
student42
gemartin@student42:~$
```

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

lsblk

```
gemartin@gemartin42:~$ lsblk
NAME                                MAJ:MIN RM  SIZE RO TYPE  MOUNTPOINT
sda                                8:0    0   30G  0 disk
├─sda1                            8:1    0   476M  0 part  /boot
├─sda2                            8:2    0     1K  0 part
├─sda5                            8:5    0  29.5G  0 part
│   └─sda5_crypt                 254:0    0  29.5G  0 crypt
│       ├─LVMGroup-root          254:1    0   9.3G  0 lvm    /
│       ├─LVMGroup-swap          254:2    0   2.1G  0 lvm    [SWAP]
│       ├─LVMGroup-home          254:3    0   4.7G  0 lvm    /home
│       ├─LVMGroup-var           254:4    0   2.8G  0 lvm    /var
│       ├─LVMGroup-srv           254:5    0   2.8G  0 lvm    /srv
│       ├─LVMGroup-tmp           254:6    0   2.8G  0 lvm    /tmp
│       └─LVMGroup-var--log      254:7    0   3.7G  0 lvm    /var/log
sr0                               11:0    1  1024M  0 rom
```

12 • Comprobar que sudo está instalado.

which sudo

```
gemartin@gemartin42:~$ which sudo
/usr/bin/sudo
gemartin@gemartin42:~$ _
```

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

```
gemartin@gemartin42:~$ dpkg -s sudo
Package: sudo
Status: install ok installed
Priority: optional
Section: admin
Installed-Size: 4589
Maintainer: Sudo Maintainers <sudo@packages.debian.org>
Architecture: amd64
Version: 1.9.5p2-3
Replaces: sudo-ldap
Depends: libaudit1 (>= 1:2.2.1), libc6 (>= 2.27), libpam0g (>= 0.99.7.1), libselinux1 (>= 3.1~), zlib1g (>= 1:1.2.0.2), libpam-modules, lsb-base
Conflicts: sudo-ldap
Conffiles:
 /etc/init.d/sudo 1153f6e6fa7c0e2166779df6ad43f1a8
 /etc/pam.d/sudo 85da64f888739f193fc0fa896680030e
 /etc/sudo.conf cdb3df319152dbf3a1ccab9d5bd01ad0
 /etc/sudo_logsrvd.conf 8f2d34058527c9b8155de178aacff2cd
 /etc/sudoers b1f89c8342752a2a29bc5a3f8fd70437
 /etc/sudoers.d/README 8d3cf36d1713f40a0ddc38e1b21a51b6
Description: Provide limited super user privileges to specific users
 Sudo is a program designed to allow a sysadmin to give limited root
 privileges to users and log root activity. The basic philosophy is to give
 as few privileges as possible but still allow people to get their work done.
.
This version is built with minimal shared library dependencies, use the
sudo-ldap package instead if you need LDAP support for sudoers.
Homepage: https://www.sudo.ws/
gemartin@gemartin42:~$
```

13 ◦ Introducimos el nuevo usuario dentro del grupo sudo.

`sudo adduser name_user sudo`

```
root@gemartin42:/home/gemartin# sudo adduser newuser sudo
Adding user `newuser' to group `sudo' ...
Adding user newuser to group sudo
Done.
root@gemartin42:/home/gemartin# _
```

Comprobamos que está dentro del grupo.

```
root@gemartin42:/home/gemartin# getent group sudo
sudo:x:27:gemartin,newuser
root@gemartin42:/home/gemartin# _
```

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

```
root@gemartin42:/var/log/sudo# nano /etc/sudoers.d/sudo_config
```

```
GNU nano 5.4 /etc/sudoers.d/sudo_config
Defaults passwd_tries=3
Defaults badpass_message="Clave incorrecta"
Defaults logfile="/var/log/sudo_config"
Defaults log_input, log_output
Defaults iolog_dir="/var/log/sudo"
Defaults requiretty
Defaults secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin"
```

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.

```
root@gemartin42:/var/log/sudo# cd
root@gemartin42:~# cd /var/log/sudo
root@gemartin42:/var/log/sudo# ls
00 seq sudo_config
root@gemartin42:/var/log/sudo#
```

```
root@gemartin42:/var/log/sudo# cat sudo_config
Nov 24 05:16:28 : root : TTY=tty1 ; PWD=/var/log/sudo ; USER=root ; TSID=00001W
; COMMAND=/usr/bin/nano 00
Nov 24 05:17:21 : root : TTY=tty1 ; PWD=/var/log/sudo ; USER=root ; TSID=00001X
; COMMAND=/usr/bin/nano hellogithub
root@gemartin42:/var/log/sudo#
```

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

```
root@gemartin42:/var/log/sudo# sudo nano hello42world_
```

```
root@gemartin42:/var/log/sudo# cat sudo_config
Nov 24 05:16:28 : root : TTY=tty1 ; PWD=/var/log/sudo ; USER=root ; TSID=00001W
; COMMAND=/usr/bin/nano 00
Nov 24 05:17:21 : root : TTY=tty1 ; PWD=/var/log/sudo ; USER=root ; TSID=00001X
; COMMAND=/usr/bin/nano hellogithub
Nov 24 05:23:10 : root : TTY=tty1 ; PWD=/var/log/sudo ; USER=root ; TSID=00001Y
; COMMAND=/usr/bin/nano hello42world
root@gemartin42:/var/log/sudo# _
```

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

`dpkg -s ufw`

```
root@gemartin42:~# dpkg -s ufw
Package: ufw
Status: install ok installed
Priority: optional
Section: admin
Installed-Size: 837
Maintainer: Jamie Strandboge <jamie@ubuntu.com>
Architecture: all
Version: 0.36-7.1
Depends: iptables, lsb-base (>= 3.0-6), ucf, python3:any, debconf (>= 0.5) | debconf-2.0
Suggests: rsyslog
Conffiles:
 /etc/default/ufw a921dd9d167380b04de4bc911915ea44
 /etc/init.d/ufw 4156943ab8a824fcf4b04cc1362eb230
 /etc/logrotate.d/ufw 12b1fb7cee76fc46f161e1ead1a22ce6
 /etc/rsyslog.d/20-ufw.conf 98e2f72c9c65ca8d6299886b524e80d1
 /etc/ufw/applications.d/ufw-bittorrent d9451245a3fb2aa85ed91533ce530f27
 /etc/ufw/applications.d/ufw-chat 73204a7a2819499d7802bc83b7e63ee9
 /etc/ufw/applications.d/ufw-directoryserver 28888bb4f7fa81ea2ca23bb86995df5b
 /etc/ufw/applications.d/ufw-dnsserver 7a2634d40515a5baab2d5b355873e1e6
 /etc/ufw/applications.d/ufw-fileserver d43adc11063000fc3c1a824071382047
 /etc/ufw/applications.d/ufw-loginsrvr 366b3845c4360ea626f78875a400446b
 /etc/ufw/applications.d/ufw-mailserver 37e7910a1da915bcf60dac1c2d157377
 /etc/ufw/applications.d/ufw-printserver 47e009dc96a9eac7b3f2c2483a889756
 /etc/ufw/applications.d/ufw-proxyserver 6e035b6921d41ae89c3d5867c593c5
 /etc/ufw/applications.d/ufw-webserver 07a41595f0b2c9865b7220bea998f8cf
 /etc/ufw/sysctl.conf 7723079fc108eda8f57eddab3079c70a
Description: program for managing a Netfilter firewall
 The Uncomplicated FireWall is a front-end for iptables, to make managing a
 Netfilter firewall easier. It provides a command line interface with syntax
 similar to OpenBSD's Packet Filter. It is particularly well-suited as a
 host-based firewall.
Homepage: https://launchpad.net/ufw
root@gemartin42:~# _
```

`sudo service ufw status`

```
root@gemartin42:~# sudo service ufw status
• ufw.service - Uncomplicated firewall
   Loaded: loaded (/lib/systemd/system/ufw.service; enabled; vendor preset: enabled)
   Active: active (exited) since Thu 2022-11-24 03:49:57 CET; 1h 35min ago
     Docs: man:ufw(8)
   Process: 315 ExecStart=/lib/ufw/ufw-init start quiet (code=exited, status=0/SUCCESS)
    Main PID: 315 (code=exited, status=0/SUCCESS)
       CPU: 43ms

Nov 24 03:49:57 gemartin42 systemd[1]: Finished Uncomplicated firewall.
Warning: journal has been rotated since unit was started, output may be incomplete.
root@gemartin42:~# _
```

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

`sudo ufw status numbered`

```
root@gemartin42:~# sudo ufw status numbered
Status: active

    To Action From
    --
[ 1] 4242 ALLOW IN Anywhere
[ 2] 80 ALLOW IN Anywhere
[ 3] 4242 (v6) ALLOW IN Anywhere (v6)
[ 4] 80 (v6) ALLOW IN Anywhere (v6)

root@gemartin42:~#
```

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

```
root@gemartin42:~# sudo ufw allow 8080
Rule added
Rule added (v6)
root@gemartin42:~# _
```

sudo ufw status numbered

```
root@gemartin42:~# sudo ufw status numbered
Status: active

      To Action From
      --
[ 1] 4242 ALLOW IN Anywhere
[ 2] 80 ALLOW IN Anywhere
[ 3] 8080 ALLOW IN Anywhere
[ 4] 4242 (v6) ALLOW IN Anywhere (v6)
[ 5] 80 (v6) ALLOW IN Anywhere (v6)
[ 6] 8080 (v6) ALLOW IN Anywhere (v6)

root@gemartin42:~# _
```

Para borrar la regla debemos utilizar el comando sudo ufw delete num_rule

```
root@gemartin42:~# sudo ufw delete 3
Deleting:
allow 8080
Proceed with operation (y|n)? y
Rule deleted
root@gemartin42:~# _
```

Comprobamos que se ha eliminado y vemos el numero de la siguiente regla que hay que borrar.

```
root@gemartin42:~# sudo ufw status numbered
Status: active

      To Action From
      --
[ 1] 4242 ALLOW IN Anywhere
[ 2] 80 ALLOW IN Anywhere
[ 3] 4242 (v6) ALLOW IN Anywhere (v6)
[ 4] 80 (v6) ALLOW IN Anywhere (v6)
[ 5] 8080 (v6) ALLOW IN Anywhere (v6)

root@gemartin42:~# _
```

Borramos de nuevo la regla.

```
root@gemartin42:~# sudo ufw delete 5
Deleting:
  allow 8080
Proceed with operation (y|n)? y
Rule deleted (v6)
root@gemartin42:~#
```

Comprobamos que solo nos quedan las reglas requeridas en el subject.

```
root@gemartin42:~# sudo ufw status numbered
Status: active

      To Action From
      --
[ 1] 4242 ALLOW IN Anywhere
[ 2] 80 ALLOW IN Anywhere
[ 3] 4242 (v6) ALLOW IN Anywhere (v6)
[ 4] 80 (v6) ALLOW IN Anywhere (v6)

root@gemartin42:~# _
```

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

which ssh

```
root@gemartin42:~# which ssh
/usr/bin/ssh
```

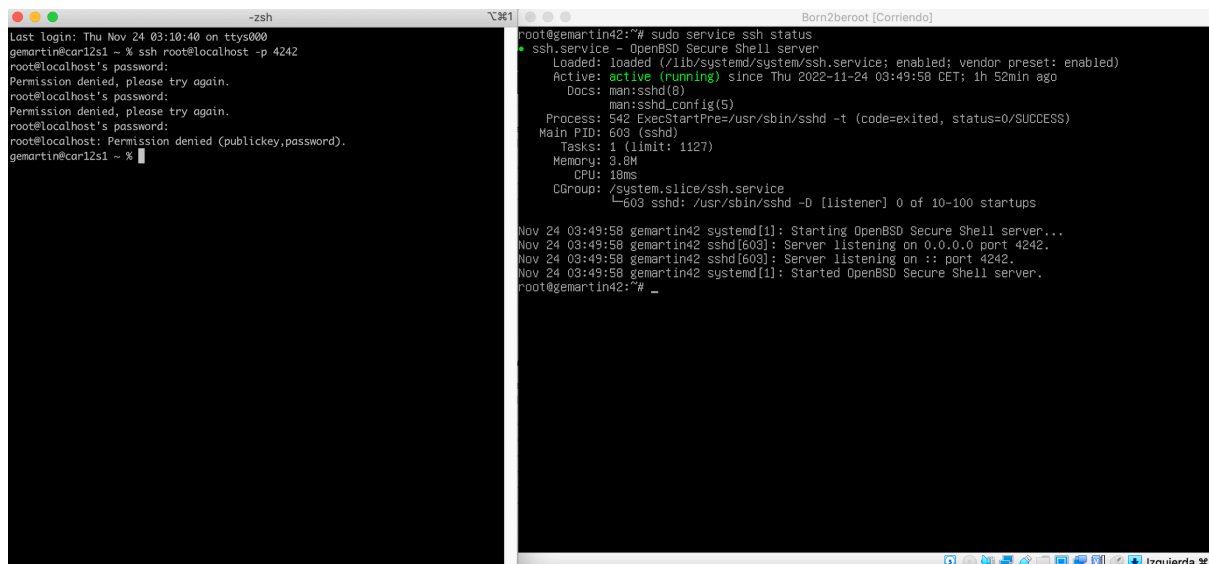
sudo service ssh status

```
root@gemartin42:~# sudo service ssh status
• ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
  Active: active (running) since Thu 2022-11-24 03:49:58 CET; 1h 48min ago
    Docs: man:sshd(8)
           man:sshd_config(5)
  Process: 542 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
 Main PID: 603 (sshd)
    Tasks: 1 (limit: 1127)
  Memory: 3.8M
     CPU: 18ms
  CGroup: /system.slice/ssh.service
          └─603 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

Nov 24 03:49:58 gemartin42 systemd[1]: Starting OpenBSD Secure Shell server...
Nov 24 03:49:58 gemartin42 sshd[603]: Server listening on 0.0.0.0 port 4242.
Nov 24 03:49:58 gemartin42 sshd[603]: Server listening on :: port 4242.
Nov 24 03:49:58 gemartin42 systemd[1]: Started OpenBSD Secure Shell server.
root@gemartin42:~#
```

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.



The image shows two terminal windows side-by-side. The left window shows an SSH connection attempt from a host named 'gemartin42' to 'localhost' on port 4242. The user 'root' is specified, but the connection is denied because 'root' is not a valid user on the target system. The right window shows the 'ssh' service status, which is 'active (running)'. It also shows the service logs, confirming that the server is listening on port 4242.

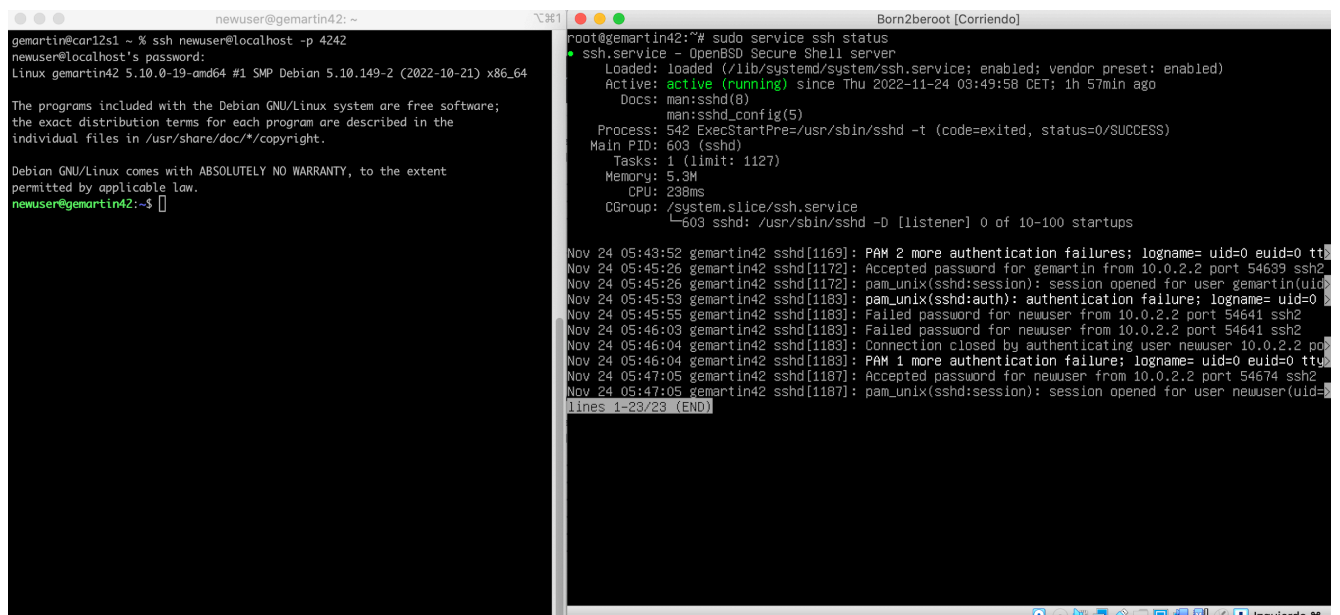
```
gemartin42@car12s1 ~ % ssh root@localhost -p 4242
root@localhost's password:
Permission denied, please try again.
root@localhost's password:
Permission denied, please try again.
root@localhost's password:
Permission denied (publickey,password).
gemartin42@car12s1 ~ %
```

```
root@gemartin42:~# sudo service ssh status
• ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
  Active: active (running) since Thu 2022-11-24 03:49:58 CET; 1h 52min ago
    Docs: man:sshd(8)
           man:sshd_config(5)
  Process: 542 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
 Main PID: 603 (sshd)
    Tasks: 1 (limit: 1127)
  Memory: 3.8M
     CPU: 18ms
  CGroup: /system.slice/ssh.service
          └─603 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

Nov 24 03:49:58 gemartin42 systemd[1]: Starting OpenBSD Secure Shell server...
Nov 24 03:49:58 gemartin42 sshd[603]: Server listening on 0.0.0.0 port 4242.
Nov 24 03:49:58 gemartin42 sshd[603]: Server listening on :: port 4242.
Nov 24 03:49:58 gemartin42 systemd[1]: Started OpenBSD Secure Shell server.
root@gemartin42:~#
```


Nos conectamos por SSH con el nuevo usuario con el comando

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



```
newuser@gemartin42: ~
gemartin@carl2s1 ~ % ssh newuser@localhost -p 4242
newuser@localhost's password:
Linux gemartin42 5.10.0-19-amd64 #1 SMP Debian 5.10.149-2 (2022-10-21) 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.
newuser@gemartin42:~$

root@gemartin42:~# sudo service ssh status
ssh.service - OpenBSD Secure Shell server
Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
Active: active (running) since Thu 2022-11-24 03:49:58 CET; 1h 57min ago
Docs: man:sshd(8)
      man:sshd_config(5)
Process: 542 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
Main PID: 603 (sshd)
Tasks: 1 (limit: 1127)
Memory: 5.3M
CPU: 238ms
CGroup: /system.slice/ssh.service
        └─603 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

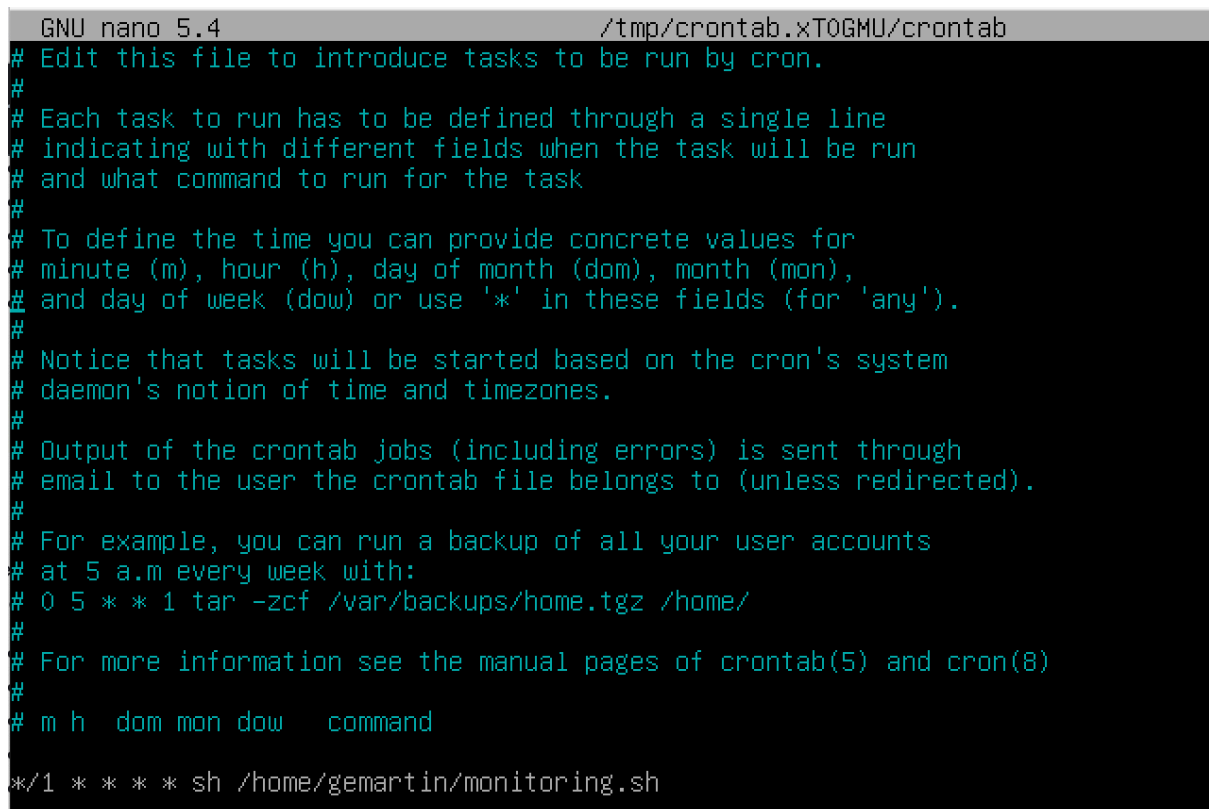
Nov 24 05:43:52 gemartin42 sshd[1169]: PAM 2 more authentication failures; logname= uid=0 euid=0 tty=
Nov 24 05:45:26 gemartin42 sshd[1172]: Accepted password for gemartin from 10.0.2.2 port 54639 ssh2
Nov 24 05:45:26 gemartin42 sshd[1172]: pam_unix(sshd:session): session opened for user gemartin(uid=0)
Nov 24 05:45:53 gemartin42 sshd[1183]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0
Nov 24 05:45:55 gemartin42 sshd[1183]: Failed password for newuser from 10.0.2.2 port 54641 ssh2
Nov 24 05:46:03 gemartin42 sshd[1183]: Failed password for newuser from 10.0.2.2 port 54641 ssh2
Nov 24 05:46:04 gemartin42 sshd[1183]: Connection closed by authenticating user newuser 10.0.2.2 port 54641
Nov 24 05:46:04 gemartin42 sshd[1183]: PAM 1 more authentication failure; logname= uid=0 euid=0 tty=
Nov 24 05:47:05 gemartin42 sshd[1187]: Accepted password for newuser from 10.0.2.2 port 54674 ssh2
Nov 24 05:47:05 gemartin42 sshd[1187]: pam_unix(sshd:session): session opened for user newuser(uid=0)
lines 1-23/23 (END)
```

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
```

```
root@gemartin42:/home/gemartin# sudo crontab -u root -e
```

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



```
GNU nano 5.4 /tmp/crontab.xTOGMU/crontab
# 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
*/1 * * * * sh /home/gemartin/monitoring.sh
```

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
```

```
root@gemartin42:/home/gemartin# sudo /etc/init.d/cron stop
Stopping cron (via systemctl): cron.service.
root@gemartin42:/home/gemartin# _
```

Si queremos que vuelva a ejecutarse:

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

```
root@gemartin42:/home/gemartin# sudo /etc/init.d/cron start
Starting cron (via systemctl): cron.service.
root@gemartin42:/home/gemartin# _
```