BORN2BEROOT

VIRTUALBOX

Name:	Born2beRoot	**Location:**	/sgoinfre
Type:	Linux	**RAM:**	1024 MB
Version:	Debian (64-bit)	**HDD:**	30 GB (Dinamic)

Settings Storage → IDE Controller → Select image (.ISO) → Start

PARTITIONS

Install → English → Other → Europe → Spain → United States (en_US.UTF-8) → American English
Hostname: '**[user_name]42**' → Domain blank → '**[root_password]**'
Name and User: '**[user_name]**' → '[**user_password]**' → Madrid → Manual '**sda**' → Yes
'**pri/log**' → Create a new partition → New primary partition of '**500m**' at '**beginning**' with mount point '**/boot**'
'**pri/log**' → Create a new partition → New logical partition of '**max**' with '**Do not mount it**'
Configure encrypted volumes → Yes → Create encrypted volumes '**sda5**' → Done, Finish, Yes and Cancel (random data) → '**[disk_password]**'
Configure the Logical Volume Manager → Yes → Create Volume Group '**LVMG**' → '**sda5**'
Create logical volume → '**LVMG**' → See table → Finish → Set mount points → See table → Finish → Yes

root	**10g**	**/**	**srv**	**3g**	**/srv**
swap	**2.3g**	**/swap**	**tmp**	**3g**	**/tmp**
home	**5g**	**/home**	**var-log**	**4g**	**Manually (/var/log)**
var	**3g**	**/var**			

Scanning Media '**No**' → Spain → deb.debian.org → No proxy → Popularity Contest '**No**'
Unselect everything → GRUB '**Yes**' → '**sda**' → Continue

Sudo

su	Login as root
apt install sudo	Install '**sudo**'
sudo reboot	Restart the system
sudo -V	Check '**sudo**' status

Groups

sudo adduser [user_name]	Add new user
sudo addgroup user42	Add new group '**user42**'
sudo adduser [user_name] sudo	Add user to group '**sudo**'
sudo adduser [user_name] user42	Add user to group '**user**

PASSWORD (sudo)

mkdir /var/log/sudo — Folder to store log of sudo commands
nano /etc/sudoers.d/sudo_config — New file

Defaults [TAB] passwd_tries=3	Max number of tries for incorrect password
Defaults [TAB] badpass_message="!!! BAD PASSWORD !!!"	Incorrect password message
Defaults [TAB] logfile="/var/log/sudo/sudo_config"	Folder to store log of sudo commands
Defaults [TAB] log_input, log_output	Log input & output commands
Defaults [TAB] iolog_dir="/var/log/sudo"	Log input & output commands
Defaults [TAB] requiretty	Require a terminal (TTY) to use '**sudo**'
Defaults [TAB] secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin"	Directories for '**sudo**' commands

PASSWORD (users)

nano /etc/login.defs — Edit file

PASS_MAX_DAYS 30	Number of days until password expiration
PASS_MIN_DAYS 2	Minimum days before password change is allowed

sudo apt install libpam-pwquality — Install library for password quality
nano /etc/pam.d/common-password — Edit file and add after '**retry=3**'

minlen=10	Minimum number of characters the password must contain
ucredit=-1	Must contain at least one uppercase letter
dcredit=-1	Must contain at least one digit
lcredit=-1	Must contain at least one lowercase letter
maxrepeat=3	Cannot have the same character repeated more than 3 times
reject_username	Cannot contain the username
difok=7	Must have at least 7 characters different from the old password
enforce_for_root	Implement this policy for the root user

BORN2BEROOT

SSH

It's the name of a protocol. Its main function being remote access to a server through a secure channel

sudo apt update	Update repositories **(/etc/apt/sources.list**)
sudo apt install openssh-server	Install SSH server
nano /etc/ssh/sshd_config	Edit file
Port 4242	Set '**SSH**' to use port '**4242**'
PermitRootLogin no	Disable direct root login for '**SSH**'
nano /etc/ssh/ssh_config	Edit file
Port 4242	Set '**SSH**' to use port '**4242**'
sudo service ssh restart	Restart server
sudo service ssh status	Service status and ports

Connect to SSH

In VirtualBox → Settings → Network → Advanced → Port Forwarding → New Rule → Host and Guest ports '**4242**'
In a terminal: **ssh [user_name]@localhost -p 4242**

UFW

It's a firewall that uses the command line

sudo apt install ufw	Install '**UFW**'
sudo ufw enable	Enable '**UFW**'
sudo ufw allow 4242	Open port '**4242**'
sudo ufw status	Service status and ports

SCRIPT

It's a sequence of commands saved in a file that, when executed, performs the function of each command

CPU	**grep "physical id" /proc/cpuinfo | wc -l**	Number of physical cores
	grep processor /proc/cpuinfo | wc -l	Number of logical cores
RAM	**free --mega | awk '$1 == "Mem:" {print $3}'**	Get the RAM used by the system
	free --mega | awk '$1 == "Mem:" {print $2}'	Get the total RAM of the system
	free --mega | awk 'Mem:" == $1 {printf(%.2f%%", ($3 * 100) / $2)}'	Get the percentage of used RAM
HDD		
	df -m | grep "/dev/" | grep -v "/boot" | awk '{hdd_use += $3} END {printf(hdd_use)}'	
	df -m | grep "/dev/" | grep -v "/boot" | awk '{hdd_total += $2} END {printf("%.0f\n"), hdd_total / 1024}'	
	df -m | grep "/dev/" | grep -v "/boot" | awk '{hdd_use += $3} {hdd_total += $2} END {printf("(%d%%)\n"), (hdd_total * 100 / hdd_use)}'	
Kernel	**uname -a**	Show OS architecture and kernel version
System	**vmstat 1 4 | tail -1 | awk '{printf($15)}'**	Show system statistics
Boot	**who -b | awk '$1 == "system" {printf($3 " " $4)}'**	Show last system boot time
LVM	**if [$(lsblk | grep "lvm" | wc -l) -gt 0]; then echo yes; else echo no; fi**	Show if '**LVM**' is active
Net	**ss -ta | grep ESTAB | wc -l**	Show active connections
Users	**users | wc -w**	Show the number of users on the system
IP	**hostname -I**	Show local IP address
MAC	**ip link | grep "link/ether" | awk '{printf($2)}'**	Show MAC address
Sudo	**journalctl _COMM=sudo | grep COMMAND | wc -l)**	Show the number of commands executed with '**sudo**'
nano /home/[user_name]/monitoring.sh		Create file '/**home/[user_name]/monitoring.sh**'
chmod +x /home/[user_name]/monitoring.sh		Set execution permissions

CRONTAB

It's a background process manager. Processes will be executed according to the '**crontab**' file

sudo chmod +x /home/[user_name]/monitoring.sh	Set execution permissions for the script
sudo crontab -u root -e	Edit the file and add
***/10 * * * * sh /home/[user_name]/monitoring.sh**	Execute the script every 10 minutes

---

SIGNATURE

Shut down the virtual machine	
Execute 'shasum Born2beRoot.vdi'	Obtain the signature of the virtual machine
Save the result in 'signature.txt'	Save the signature and upload it to the repository
Clone the virtual machine	**!!! DO NOT OPEN THE ORIGINAL MACHINE !!! only the cloned**

BORN2BEROOT

LIGHTTPD

It's a web server designed to be fast, secure, and flexible

sudo apt install lighttpd	Install '**LightTPD**'
sudo ufw allow 80	Open port '**80**' in '**UFW**'
sudo uwf status	Check that port '**80**' has been opened

MARIADB

It's a database

sudo apt install mariadb-server	Install '**MariaDB**'	
sudo mysql_secure_installation	Script to restrict access to the server	
Switch to unix_socket autentication? → N		
Change the root password? → N		
Remove anonymous users? → Y		
Disallow root login remotely? → Y		
Remove test database and access to it? → Y		
Reload privilege tables now? → Y		
mariadb	Run '**MariaDB**'	
CREATE DATABASE [db_name];	Create a database for '**WordPress**'	(Use only alphanumeric chars)
SHOW DATABASES;	Check that it has been created correctly	
CREATE USER '[db_user_name]'@'localhost' IDENTIFIED BY '[db_user_password]';	Create a user	(Use only alphanumeric chars)
GRANT ALL PRIVILEGES ON [db_name].* TO '[db_user_name]'@'localhost';	Link user to the database	(Use only alphanumeric chars)
FLUSH PRIVILEGES;	Update permissions	
exit	Exit '**MariaDB**'	

WORD PRESS

It's web platform for site creation and management, featuring themes and plugins for customization

sudo apt install wget zip	Installs the prerequisites '**wget**' and '**zip**'
cd /var/www	Changes the directory to '**/var/www**'
sudo wget https://es.wordpress.org/latest-es_ES.zip	Downloads '**WordPress**'
sudo unzip latest-es_ES.zip	Unzips the '**WordPress**' file
sudo mv html /html_bak	Renames '**/var/www/html**'
sudo mv /wordpress /html	Renames the '**WordPress**' folder to 'html'
sudo chmod -R 755 html	Sets permissions for the folder
localhost	Access '**WordPress**' at '**localhost**' in a web browser

Fill in the details:

Site Title:	**[title_name]**	**Password:**	**[password]**
Username:	**[user_name]**	**Email:**	**[user_name]@student.42malaga.com**
Install WordPress			

localhost/wp-admin	To access the '**WordPress**' admin panel

PHP

It's a programming language used to develop dynamic web applications and interactive websites

sudo apt install php-cgi php-mysql			Install packages and requirements
cd /var/www/html			Change to '**/var/www/html**' directory
cp wp-config-sample.php wp-config.php			Copy '**wp-config-sample.php**' file
nano wp-config.php			Edit and modify
'database_name_here'	→	**'[db_name]'**	Same as '**MariaDB**' '**[db_name]**'
'username_here'	→	**'[db_user_name]'**	Same as '**MariaDB** '**[db_user_name]**'
'password_here'	→	**'[db_user_password]'**	Same as '**MariaDB**' '**[db_user_password]**'
sudo lighty-enable-mod fastcgi			Enable '**fastcgi**' module in '**LightTPD**'
sudo lighty-enable-mod fastcgi-php			Enable '**fastcgi-php**' module in '**LightTPD**'
sudo service lighttpd force-reload			Update and apply changes

COCKPIT

It's a web-based systems management interface

sudo apt update	Updates available packages
sudo apt install cockpit	Installs '**Cockpit**'
sudo systemctl start cockpit	Starts '**Cockpit**' service
sudo systemctl enable cockpit.socket	Enables '**Cockpit**' to start on boot
sudo ufw allow 9090	Opens port '**9090**' in the '**UFW**'
localhost:9090	Access '**Cockpit**' at '**localhost:9090**' in a web browser

SCRIPT

Script to be used. Copy to '**/home/[user_name]/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 += $2} END {printf ("%.1fGb\n"), disk_t/1024}')
disk_use=$(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 CONECTIONS
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"
```

TESTER

```
cd /home/[user_name]/                                  Go to the user's folder
sudo apt update                                        Update packages
sudo apt git                                           Install 'git'
git clone https://github.com/gemartin99/Born2beroot-Tester.git   Download 'Tester'
cd Born2beroot-Tester/                                 Go to the 'Tester' folder
bash Test.sh                                           Execute the 'Tester'
```

BORN2BEROOT

EVALUACIÓN

Como funciona 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.
Porque he elegido '**Debian**'	Desde 1993, filosofía de software libre y gran comunidad. Soporte para muchas arquitecturas de hardware. Grandes repositorios. Mucha documentación, más fácil para nuevos usuarios.
Las diferencias básicas entre '**Rocky**' y '**Debian**'	Debian es una distribución Linux enfocado en el software libre, con múltiples arquitecturas y versiones estables regulares. Rocky apunta a entornos empresariales, priorizando estabilidad y compatibilidad.
El objetivo de las máquinas virtuales	Su objetivo es el de proporcionar un entorno de ejecución independiente de un sistema operativo y que permita que un programa se ejecute siempre de la misma forma sobre cualquier plataforma.
La diferencia entre '**Aptitude**' y '**APT**'	APT es un administrador de paquetes de nivel inferior y Aptitude es un administrador de paquetes de alto nivel. Aptitude ofrece una mejor funcionalidad en comparación con APT. Ambos son capaces de gestionar paquetes. Aptitude es una versión mejorada de APT.
Que es '**APPArmor**'	Es un módulo de seguridad del kernel Linux que permite al administrador del sistema restringir las capacidades de un programa.

Iniciar la máquina virtual			
Comprobar que no tiene interfaz gráfica	**ls /usr/bin/*session**		
Contraseña del encriptado	**[disk_password]**		
Iniciar sesión	**[user_name]**	**[user_password]**	(l-u-d-!3c-!user-dif7-min10)
Comprobar '**UFW**'	**sudo ufw status**		**sudo service ufw status**
Comprobar '**SSH**'	**sudo service ssh status**		
Comprobar versión del sistema '**Debian**'	**uname -v**		

Comprobar que usuario es mi login y que pertenece a '**sudo**' y '**user42**'	**getent group sudo**	**getent group user42**
Crear nuevo usuario	**sudo adduser [eval_user_name]**	**[eval_user_password]**
Crear un grupo llamado '**evaluating**'	**sudo addgroup evaluating**	
Asignar nuevo usuario al grupo	**sudo adduser [eval_user_name] evaluating**	
Comprobar que pertenece correctamente	**getent group evaluating**	
Explicar las ventajas y desventajas de la password policy		

Explicar '**LVM**' (Logical Volume Manager)	Es un gestor de volúmenes lógicos. Proporciona un método para asignar espacio en dispositivos de almacenamiento. Es más flexible que las particiones convencionales.

Comprobar el '**hostname**'	**hostname**		
Modifica el '**hostname**' por '**[eval_user_name]42**' y reinicia	**sudo nano /etc/hosts**	**sudo nano /etc/hostname**	**sudo reboot**
Comprobar que '**hostname**' ha cambiado	**hostname**		
Restaura el '**hostname**' por '**[user_name]42**' y reinicia	**sudo nano /etc/hosts**	**sudo nano /etc/hostname**	**sudo reboot**
Mostrar las particiones	**lsblk**		

Explicar '**Sudo**'	Sudo permite ejecutar comandos con privilegios de administrador. Otorga temporalmente permisos elevados para instalar un software o modificar archivos.

Comprobar que '**sudo**' está instalado	**wich sudo**	**dpkg -s sudo**
Asignar el nuevo usuario al grupo '**sudo**'	**sudo adduser [eval_user_name] sudo**	**getent group sudo**
Mostrar reglas de sudo	**cat /etc/sudoers.d/sudo_config**	
Comprobar '**/var/log/sudo**' existe y tiene el ultimo comando	**sudo ls /var/log/sudo**	**sudo cat /var/log/sudo/sudo_config**

Explicar '**UFW**' (Uncomplicated FireWall)	Un firewall es una barrera de seguridad que regula el tráfico de red. Filtra el tráfico según reglas predefinidas.

Comprueba que '**UFW**' está instalado	**dpkg -s ufw**	**sudo service ufw status**
Mostrar las reglas de '**UFW**'	**sudo ufw status numbered**	
Añade una regla para el puerto '**8080**'	**sudo ufw allow 8080**	**sudo ufw status numbered**
Elimina la regla para el puerto '**8080**'	**sudo ufw delete [num_rule]**	**sudo ufw status numbered**

BORN2BEROOT

Explicar '**SSH**' (Secure Shell) — SSH es un protocolo de red cifrado que proporciona una forma segura de acceder a sistemas remotos.

Comprueba que '**SSH**' está instalado y usa el puerto '**4242**'	**which ssh**	**sudo service ssh status**
Conectar con root (debe fallar)	**en iTerm: ssh root@localhost -p 4242**	**[root_password]**
Conectar con '**[user_name]**'	**en iTerm: ssh [eval_user_name]@localhost -p 4242**	**[eval_user_password]**

Explicar '**Cron**' (Chronos) — Cron es un programa de administración de tareas que permite programar la ejecución automática de comandos, scripts o procesos.

Cambiar '**Cron**' para que se ejecute cada minuto	**sudo crontab -u root -e**	
Detener y Reiniciar	**sudo /etc/init.d/cron stop**	**sudo /etc/init.d/cron start**

Explicar '**Cockpit**' — Cockpit es una interfaz web que permite monitorear y administrar servidores y sistemas a través de un navegador web.

Mostrar las particiones	**lsblk**		
Mostrar '**LightTPD**'	**dpkg -l | grep lighttpd**		**systemctl status lighttpd**
Mostrar '**MariaDB**'	**dpkg -l | grep mariadb**		**systemctl status mariadb**
Mostrar '**WordPress**'	**s /var/www/html/**		En browser: **localhost**
Mostrar '**Cockpit**'	**dpkg -s cockpit**	**systemctl status cockpit**	En browser: **localhost:9090**

Explicar '**Script**' — Es una secuencia de comandos guardados en un archivo. Cuando se ejecuta, realiza la función de cada uno de los comandos.

#!/BIN/BASH	Indica que interprete de comandos se debe utilizar
UNAME	Información del sistema
FREE	Cantidad de memoria libre y utilizada
DF	Espacio libre y utilizado en sistemas de archivos montados
CMSTAT	Muestra información del uso de memoria, actividad del sistema, CPU y procesos
EXPR	Permite evaluar expresiones (aritméticas o de cadenas)
PRINT/PRINTF	**C**omandos para imprimir texto en pantalla
WHO	Información sobre usuarios conectados al sistema
LSBLK	Lista información sobre dispositivos de almacenamiento
SS	Información sobre sockets del sistema (conexiones de red)
USERS	Nombres de usuario que están conectados
HOSTNAME	Muestra o establece el nombre del sistema
IP LINK	Información sobre las interfaces de red disponibles
JOURNALCTL	Visualización y consulta de registros del sistema
GREP	Filtra líneas de texto que coinciden con un patrón
AWK	Herramienta para procesamiento de texto, búsqueda y extracción de patrones
WC	Cuenta líneas, palabras y bytes en archivos de texto
PIPE '|'	Operador para redirigir la salida de un comando hacia la entrada de otro
WALL	Envía un mensaje a todos los usuarios conectados, mostrándolo como una notificación en sus terminales

TESTER

cd /home/[user_name]/	Ir a la carpeta del usuario
sudo apt update	Actualizar los paquetes
sudo apt git	Instalar '**git**'
git clone https://github.com/gemartin99/Born2beroot-Tester.git	Descargar '**Tester**'
cd Born2beroot-Tester/	Ir a la carpeta '**Tester**'
bash Test.sh	Ejecutar '**Tester**'