IMPLEMENTO BIM 2.0 · SERIE TÉCNICA

UBUNTU BLINDADO

Aprende Linux desde cero
y protege tu sistema

Terminal · Permisos · Redes · Cortafuegos · SSH
Auditoría · Copias de seguridad · Respuesta a incidentes

MARLON FALCÓN HERNÁNDEZ

Ubuntu Blindado

Aprende Linux desde cero y protege tu sistema

© 2026 Marlon Falcón Hernández · Implemento BIM 2.0
Primera edición · Valencia, octubre de 2026

Ubuntu es una marca registrada de Canonical Ltd. Esta obra es independiente y no está patrocinada ni avalada por Canonical. Los comandos se ofrecen con fines formativos: pruébalos primero en un entorno de laboratorio.

Índice

Prólogo: cómo usar este libro 4

PARTE I · APRENDE LINUX

01 Qué es Linux y qué es Ubuntu 6

02 Instalar Ubuntu de forma segura 8

03 La terminal y los comandos esenciales 10

04 Sistema de archivos y permisos 12

05 Usuarios, grupos y sudo 14

06 Paquetes y actualizaciones 16

07 Procesos, servicios y systemd 18

08 Redes en Ubuntu 20

PARTE II · PROTEGE TU SISTEMA

09 Fundamentos de seguridad y bastionado 23

10 Cortafuegos con UFW 25

11 SSH seguro 27

12 Fail2ban, AppArmor y auditoría 29

13 Copias de seguridad y cifrado 31

14 Monitorización, logs y respuesta a incidentes 33

15 Proyecto final: un servidor de producción blindado 35

Apéndice. Chuleta de comandos 37

PRÓLOGO

Prólogo: cómo usar este libro

Este libro te lleva de no haber abierto nunca una terminal a administrar un servidor Ubuntu con criterios de seguridad profesionales. Está pensado para técnicos, desarrolladores, consultores ERP y cualquiera que despliegue aplicaciones en Linux.

La obra tiene dos partes. Los capítulos 1 a 8 enseñan Linux: instalación, terminal, archivos, usuarios, paquetes, servicios y redes. Los capítulos 9 a 15 enseñan a protegerlo: bastionado, cortafuegos, SSH, detección de intrusos, cifrado, copias y respuesta a incidentes.

CÓMO LEERLO

- Practica en una máquina virtual (VirtualBox, VMware, Multipass o un VPS barato), nunca en un servidor en producción.
- Escribe los comandos a mano; copiarlos no deja memoria muscular.
- Cada capítulo termina con ejercicios. Hazlos antes de pasar al siguiente.
- Los bloques que empiezan por `$` se ejecutan como usuario normal; los que llevan `sudo` necesitan privilegios.

CONVENCIONES

- `texto en monoespaciado`: comandos, rutas y archivos.
- **Atención**: acciones que pueden dejarte sin acceso o borrar datos.
- Los ejemplos se han escrito para Ubuntu LTS (24.04 y posteriores). Casi todo funciona igual en Debian y derivadas.

PARTE I

Aprende Linux

CAPÍTULO
01

Qué es Linux y qué es Ubuntu

Linux es un núcleo (kernel) de sistema operativo libre, creado por Linus Torvalds en 1991. Ubuntu es una distribución: el núcleo Linux más miles de programas, un instalador y un sistema de paquetes, mantenida por la empresa Canonical desde 2004.

1.1 Piezas de un sistema Linux

1. **Kernel**: gestiona CPU, memoria, discos y red.
2. **Init (systemd)**: el primer proceso; arranca y vigila los servicios.
3. **Shell (bash)**: el intérprete donde escribes comandos.
4. **Utilidades GNU**: `ls`, `cp`, `grep` y cientos más.
5. **Entorno gráfico (GNOME)**: solo en la edición Desktop.

1.2 Versiones y ciclo de vida

Ubuntu publica dos versiones al año, en abril (`.04`) y octubre (`.10`). Cada dos años, la de abril es **LTS** (Long Term Support): recibe parches de seguridad durante 5 años, ampliables con Ubuntu Pro.

Tipo	Ejemplo	Soporte	Uso recomendado
LTS	24.04, 26.04	5 años (+ Pro)	Servidores y empresa
Intermedia	25.10	9 meses	Probar novedades

> **Regla de oro de seguridad:** en producción, usa siempre una LTS con soporte vigente. Un sistema sin parches es el primer agujero de cualquier empresa.

1.3 Ediciones

- **Desktop**: escritorio gráfico para el día a día.
- **Server**: sin entorno gráfico, menos software instalado y por tanto menos superficie de ataque.
- **Cloud / Core**: imágenes mínimas para nubes y dispositivos IoT.

EJERCICIOS

1. Averigua qué versión de Ubuntu tienes con `lsb_release -a` y `uname -r`.
2. Explica con tus palabras por qué un servidor no debería llevar entorno gráfico.

CAPÍTULO
02

Instalar Ubuntu de forma segura

Una instalación segura empieza antes de arrancar: imagen verificada, disco cifrado y solo el software necesario.

2.1 Descargar y verificar la imagen

Descarga la ISO solo desde ubuntu.com y comprueba su huella SHA-256 contra el archivo `SHA256SUMS` publicado junto a ella:

```
$ sha256sum ubuntu-24.04-live-server-amd64.iso
$ grep live-server SHA256SUMS
```

Si las dos cadenas no coinciden, la imagen está corrupta o manipulada. No la uses.

2.2 Laboratorio virtual

Para practicar, la vía más rápida es Multipass, que crea máquinas Ubuntu en segundos:

```
$ sudo snap install multipass
$ multipass launch --name lab --cpus 2 --memory 2G --disk 20G
$ multipass shell lab
```

2.3 Decisiones durante la instalación

Pantalla	Elección segura	Motivo
Disco	LVM + cifrado LUKS	Protege los datos si roban el disco
Usuario	Nombre propio, nunca `admin`	Evita ataques de diccionario
Contraseña	14+ caracteres o frase	Resiste fuerza bruta
OpenSSH	Instalar solo si es servidor remoto	Menos servicios expuestos
Snaps extra	Ninguno	Instala solo lo que necesites

2.4 Primeros cinco minutos tras instalar

```
$ sudo apt update && sudo apt full-upgrade -y
$ sudo timedatectl set-timezone Europe/Madrid
$ sudo hostnamectl set-hostname srv-lab01
$ sudo reboot
```

EJERCICIOS

1. Instala Ubuntu Server en una máquina virtual con disco cifrado.
2. Verifica una ISO con `sha256sum` y documenta el resultado.

CAPÍTULO
03

La terminal y los comandos esenciales

La terminal es la herramienta principal de un administrador: es más rápida que el ratón, se puede automatizar y funciona igual en local y por SSH.

3.1 Anatomía de un comando

```
$ ls -lh /var/log
#  |   |     └── argumento: sobre qué actúa
#  |   └──────── opciones: cómo lo hace
#  └──────────── comando
```

Para aprender cualquier comando usa `man ls`, `ls --help` o `tldr ls` (tras `sudo apt install tldr`).

3.2 Moverse y mirar

Comando	Qué hace
`pwd`	Muestra la carpeta actual
`cd /etc` · `cd ~` · `cd ..`	Cambia de carpeta
`ls -la`	Lista archivos, incluidos los ocultos
`cat`, `less`, `head -n 20`, `tail -f`	Lee archivos y sigue logs en vivo
`find / -name '*.conf' 2>/dev/null`	Busca archivos
`grep -ri 'error' /var/log`	Busca texto dentro de archivos

3.3 Crear, copiar, mover y borrar

```
$ mkdir -p proyectos/demo
$ touch proyectos/demo/notas.txt
$ cp -r proyectos copia_proyectos
$ mv notas.txt notas_antiguas.txt
$ rm -i archivo.txt
```

> **Atención:** `rm -rf` no pide confirmación ni usa papelera. Revisa la ruta dos veces antes de pulsar Intro.

3.4 Tuberías y redirecciones

La potencia de Linux está en encadenar herramientas pequeñas:

```
$ ps aux | grep nginx          # salida de uno, entrada del otro
$ df -h > discos.txt           # guarda la salida en un archivo
$ echo 'linea' >> notas.txt    # añade al final
$ comando 2> errores.log       # guarda solo los errores
```

3.5 Editar archivos

`nano` es el editor más sencillo (Ctrl+O guarda, Ctrl+X sale). `vim` es más potente y conviene conocer lo mínimo: `i` para escribir, `Esc` y `:wq` para guardar y salir.

EJERCICIOS

1. Encuentra los archivos `.log` de `/var/log` mayores de 1 MB con `find /var/log -name '*.log' -size +1M`.
2. Cuenta cuántas líneas de `/var/log/syslog` contienen la palabra `error`.

CAPÍTULO
04

Sistema de archivos y permisos

En Linux todo es un archivo, y cada archivo tiene un dueño, un grupo y unos permisos. Entender esto es la base de toda la seguridad del sistema.

4.1 Mapa de carpetas

Ruta	Contenido
`/etc`	Configuración del sistema
`/home`	Carpetas personales de los usuarios
`/root`	Carpeta del superusuario
`/var`	Datos variables: logs, bases de datos, webs
`/tmp`	Temporales, se vacía al reiniciar
`/usr`	Programas y bibliotecas
`/opt`	Software de terceros (por ejemplo, Odoo)
`/dev` , `/proc` , `/sys`	Dispositivos e información del kernel

4.2 Leer los permisos

```
$ ls -l informe.txt
-rw-r----- 1 marlon contabilidad 2048 oct 31 informe.txt
```

Los diez primeros caracteres se leen así: tipo (`-` archivo, `d` carpeta), luego tres bloques de `rwx` para **dueño**, **grupo** y **otros**. `r` = leer (4), `w` = escribir (2), `x` = ejecutar o entrar en la carpeta (1).

En el ejemplo: el dueño lee y escribe (6), el grupo solo lee (4) y el resto no tiene acceso (0). Es decir, `640` .

4.3 Cambiar permisos y dueños

```
$ chmod 640 informe.txt          # forma numérica
$ chmod u+x script.sh            # forma simbólica
$ sudo chown marlon:contabilidad informe.txt
$ sudo chmod -R 750 /opt/miapp
```

Valores habituales: `644` para archivos públicos, `600` para claves privadas, `755` para carpetas y scripts, `700` para carpetas privadas.

4.4 Permisos especiales

- **SUID** (`chmod u+s`): el programa se ejecuta con los permisos de su dueño. Es peligroso si el dueño es root; vigílalo.
- **SGID** en carpetas: los archivos nuevos heredan el grupo. Útil para carpetas compartidas.
- **Sticky bit** (`chmod +t`): en `/tmp`, cada usuario solo puede borrar lo suyo.

Para auditar binarios SUID del sistema:

```
$ sudo find / -perm -4000 -type f 2>/dev/null
```

4.5 umask y ACL

`umask` define los permisos por defecto de lo que creas; `027` es una buena opción para servidores. Si necesitas dar acceso a un usuario concreto sin tocar grupos, usa ACL: `setfacl -m u:ana:r informe.txt` y consúltalas con `getfacl`.

EJERCICIOS

1. Crea una carpeta compartida `/srv/equipo` donde todo el grupo `equipo` pueda escribir y nadie más pueda entrar.
2. Lista los binarios SUID de tu sistema y busca para qué sirve cada uno.

CAPÍTULO
05

Usuarios, grupos y sudo

El principio del mínimo privilegio dice que cada persona y cada servicio debe tener solo los permisos que necesita. En Linux se aplica con usuarios, grupos y `sudo`.

5.1 Dónde vive la información

- `/etc/passwd`: lista de usuarios (nombre, UID, carpeta, shell).
- `/etc/shadow`: contraseñas cifradas; solo root puede leerlo.
- `/etc/group`: grupos y sus miembros.

```
$ id                    # quién soy y en qué grupos estoy
$ who                   # quién está conectado
$ last -n 10            # últimos inicios de sesión
```

5.2 Gestionar usuarios

```
$ sudo adduser ana                    # crea usuario con carpeta y contraseña
$ sudo usermod -aG desarrolladores ana # añade a un grupo (¡-a es obligatorio!)
$ sudo passwd -l ana                  # bloquea la cuenta
$ sudo deluser --remove-home ana      # elimina usuario y su carpeta
```

> **Atención:** `usermod -G` sin `-a` reemplaza todos los grupos del usuario y puede quitarle el acceso a `sudo`.

5.3 sudo: privilegios con registro

Ubuntu bloquea la cuenta root por defecto. En su lugar, los usuarios del grupo `sudo` elevan privilegios comando a comando, y cada uso queda registrado en `/var/log/auth.log`.

Para reglas finas, edita siempre con `visudo` (valida la sintaxis antes de guardar):

```
$ sudo visudo -f /etc/sudoers.d/odoo
# Permite a 'deploy' reiniciar solo el servicio odoo
deploy ALL=(root) NOPASSWD: /usr/bin/systemctl restart odoo
```

5.4 Políticas de contraseñas

```
$ sudo apt install libpam-pwquality
$ sudo nano /etc/security/pwquality.conf
# minlen = 14
# minclass = 3
$ sudo chage -M 365 -W 14 ana   # caducidad y aviso previo
```

5.5 Usuarios de servicio

Cada aplicación (base de datos, servidor web, Odoo) debe ejecutarse con su propio usuario sin shell de acceso:

```
$ sudo adduser --system --group --no-create-home --shell /usr/sbin/nologin miapp
```

Si la aplicación es atacada, el intruso queda encerrado en los permisos de ese usuario.

EJERCICIOS

1. Crea un usuario `deploy` que solo pueda reiniciar un servicio concreto con `sudo`.
2. Revisa `/var/log/auth.log` y localiza tu último uso de `sudo`.

CAPÍTULO
06

Paquetes y actualizaciones

La mayoría de intrusiones explotan fallos ya corregidos que nadie instaló. Mantener el sistema actualizado es la medida de seguridad con mejor relación esfuerzo-resultado.

6.1 APT, el gestor de paquetes

```
$ sudo apt update                # refresca la lista de paquetes
$ apt list --upgradable          # qué hay pendiente
$ sudo apt upgrade               # actualiza
$ sudo apt install nginx         # instala
$ sudo apt remove --purge nginx  # desinstala y borra su configuración
$ sudo apt autoremove            # limpia dependencias huérfanas
$ apt show nginx                 # información de un paquete
```

Los repositorios se definen en `/etc/apt/sources.list.d/`. Añade repositorios de terceros solo de fuentes oficiales y con su clave GPG; cada repositorio es alguien en quien confías para ejecutar código como root.

6.2 Snap

Snap empaqueta aplicaciones con sus dependencias y las aísla del sistema. Se actualizan solas.

```
$ snap list
$ sudo snap install code --classic
$ sudo snap refresh
```

6.3 Actualizaciones automáticas de seguridad

```
$ sudo apt install unattended-upgrades
$ sudo dpkg-reconfigure -plow unattended-upgrades
```

Revisa `/etc/apt/apt.conf.d/50unattended-upgrades` para decidir si se reinicia solo (`Automatic-Reboot`) y a qué hora. Los logs están en `/var/log/unattended-upgrades/`.

6.4 Ubuntu Pro y Livepatch

Ubuntu Pro amplía los parches de seguridad al repositorio *universe* y alarga el soporte de las LTS. Es gratuito para uso personal en un número limitado de máquinas. Incluye Livepatch, que aplica parches críticos del kernel sin reiniciar.

```
$ sudo pro attach <TU_TOKEN>
$ pro status
```

6.5 ¿Hace falta reiniciar?

```
$ ls /var/run/reboot-required && cat /var/run/reboot-required.pkgs
```

EJERCICIOS

1. Activa `unattended-upgrades` y fuerza una prueba con `sudo unattended-upgrade --dry-run -d`.
2. Enumera los repositorios de terceros configurados en tu equipo y justifica cada uno.

CAPÍTULO
07

Procesos, servicios y systemd

Un proceso es un programa en ejecución; un servicio es un proceso que systemd arranca, vigila y reinicia. Saber qué se ejecuta en tu máquina es saber qué puede ser atacado.

7.1 Ver procesos

```
$ ps aux --sort=-%cpu | head     # los que más CPU consumen
$ top                            # vista en vivo (q para salir)
$ htop                           # versión más amigable
$ pgrep -a nginx                 # buscar por nombre
$ kill 1234                      # terminar con cortesía (SIGTERM)
$ kill -9 1234                   # forzar (SIGKILL), último recurso
```

7.2 Gestionar servicios con systemctl

Acción	Comando
Estado	`systemctl status ssh`
Arrancar / parar	`sudo systemctl start ssh` · `stop`
Reiniciar / recargar config	`sudo systemctl restart ssh` · `reload`
Activar al arranque	`sudo systemctl enable --now ssh`
Desactivar	`sudo systemctl disable --now cups`
Listar los activos	`systemctl list-units --type=service --state=running`

7.3 Reducir la superficie de ataque

Cada servicio activo es una puerta. Revisa cuáles corren y desactiva los que no uses (impresión, Bluetooth, avahi en servidores):

```
$ systemctl list-unit-files --state=enabled
$ sudo systemctl disable --now avahi-daemon cups bluetooth
```

7.4 Crear tu propio servicio con aislamiento

systemd puede encerrar un servicio con pocas líneas. Ejemplo en `/etc/systemd/system/miapp.service`:

```
[Unit]
Description=Mi aplicación
After=network.target

[Service]
User=miapp
ExecStart=/opt/miapp/bin/server
Restart=on-failure
NoNewPrivileges=true
ProtectSystem=strict
ProtectHome=true
PrivateTmp=true
ReadWritePaths=/var/lib/miapp

[Install]
WantedBy=multi-user.target
```

Después: `sudo systemctl daemon-reload && sudo systemctl enable --now miapp`. Puedes puntuar el aislamiento de cualquier servicio con `systemd-analyze security miapp`.

7.5 Tareas programadas

`crontab -e` edita las tareas de tu usuario. Formato: minuto, hora, día, mes, día de la semana, comando. Ejemplo, copia cada noche a las 02:30: `30 2 * * * /usr/local/bin/backup.sh`. Los timers de systemd (`systemctl list-timers`) son la alternativa moderna y registran cada ejecución.

EJERCICIOS

1. Lista los servicios activos de tu máquina y desactiva dos que no necesites.
2. Ejecuta `systemd-analyze security` y elige el servicio peor puntuado para investigar.

CAPÍTULO
08

Redes en Ubuntu

Un servidor se ataca casi siempre por la red. Antes de protegerla hay que saber leerla: interfaces, direcciones, rutas, DNS y puertos abiertos.

8.1 Diagnóstico básico

```
$ ip a                    # interfaces y direcciones IP
$ ip r                    # tabla de rutas y puerta de enlace
$ resolvectl status       # servidores DNS en uso
$ ping -c 4 1.1.1.1       # conectividad
$ tracepath ubuntu.com    # por dónde viajan los paquetes
$ curl -I https://ubuntu.com
```

8.2 Configurar la red con Netplan

Ubuntu define la red en archivos YAML de `/etc/netplan/`. Ejemplo de IP fija para un servidor:

```
network:
  version: 2
  ethernets:
    enp0s3:
      addresses: [192.168.1.50/24]
      routes:
        - to: default
          via: 192.168.1.1
      nameservers:
        addresses: [1.1.1.1, 9.9.9.9]
```

Aplica con `sudo netplan try`: si pierdes la conexión, revierte solo a los 120 segundos. Usa `try` siempre en servidores remotos.

8.3 ¿Qué puertos están escuchando?

Es la pregunta de seguridad más importante de este capítulo:

```
$ sudo ss -tulpn
```

Cada línea es un servicio aceptando conexiones. Fíjate en la dirección: `127.0.0.1:5432` solo es accesible desde la propia máquina; `0.0.0.0:5432` está abierto a toda la red. Una base de datos casi nunca debería escuchar en `0.0.0.0`.

8.4 Comprobar desde fuera

Lo que ves dentro no siempre coincide con lo que se ve desde Internet. Desde otra máquina de tu laboratorio:

```
$ sudo apt install nmap
$ nmap -sV 192.168.1.50
```

> **Atención:** escanea solo equipos propios o con autorización por escrito. Escanear sistemas ajenos puede ser delito.

8.5 Archivo hosts

`/etc/hosts` resuelve nombres antes que el DNS. Revísalo de vez en cuando: el malware lo modifica para redirigir dominios legítimos.

EJERCICIOS

1. Configura una IP fija con Netplan usando `netplan try`.
2. Ejecuta `ss -tulpn`, identifica cada proceso que escucha y decide si debe estar expuesto.

PARTE II

Protege tu sistema

CAPÍTULO
09

Fundamentos de seguridad y bastionado

Bastionar (*hardening*) es reducir todo lo que un atacante podría aprovechar. No existe el sistema invulnerable: el objetivo es que atacarte sea más caro que atacar a otro, y enterarte si alguien lo intenta.

9.1 Los cuatro principios

1. **Mínimo privilegio**: cada usuario y servicio, solo lo imprescindible.
2. **Defensa en profundidad**: varias capas; si una cae, la siguiente resiste.
3. **Mínima superficie**: lo que no está instalado no se puede explotar.
4. **Visibilidad**: lo que no se registra no se puede investigar.

9.2 Modelo de amenazas en cinco preguntas

Antes de configurar nada, responde por escrito:

- ¿Qué protejo? (datos de clientes, código, facturación)
- ¿De quién? (bots automáticos, ex empleados, competencia)
- ¿Por dónde podrían entrar? (SSH, web, correo, USB)
- ¿Qué pasa si lo pierdo? (multas RGPD, parada de negocio)
- ¿Cuánto tardaría en recuperarme?

Para la mayoría de pymes, la amenaza real son bots que prueban contraseñas y explotan software sin actualizar, no hackers dirigidos.

9.3 Ajustes del kernel (sysctl)

Crea `/etc/sysctl.d/99-hardening.conf`:

```
# Ignorar redirecciones y rutas de origen (evita ataques de red)
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.all.accept_source_route = 0
# Protección contra SYN flood y spoofing
net.ipv4.tcp_syncookies = 1
net.ipv4.conf.all.rp_filter = 1
# Registrar paquetes con direcciones imposibles
net.ipv4.conf.all.log_martians = 1
# Ocultar punteros del kernel y restringir dmesg
kernel.kptr_restrict = 2
kernel.dmesg_restrict = 1
```

Aplica con `sudo sysctl --system`.

9.4 Medir antes y después: Lynis

Lynis audita el sistema y da una puntuación de bastionado con recomendaciones concretas:

```
$ sudo apt install lynis
$ sudo lynis audit system
```

Apunta el *hardening index* inicial. Repite la auditoría al terminar el libro y compara.

9.5 Seguridad física y arranque

- Cifrado de disco completo (LUKS) en portátiles, siempre.
- Contraseña en BIOS/UEFI y arranque seguro (*Secure Boot*) activado.
- Bloqueo automático de pantalla en equipos de escritorio.

EJERCICIOS

1. Escribe el modelo de amenazas de un servidor real de tu trabajo.
2. Ejecuta Lynis y anota la puntuación y las cinco primeras sugerencias.

CAPÍTULO 10

Cortafuegos con UFW

Un cortafuegos decide qué tráfico entra y sale. La política correcta para un servidor es denegar toda entrada y abrir solo lo imprescindible. En Ubuntu, la herramienta es UFW (*Uncomplicated Firewall*), una capa sencilla sobre nftables.

10.1 Configuración base sin quedarte fuera

> **Atención:** si administras por SSH, permite SSH *antes* de activar el cortafuegos o perderás la conexión.

```
$ sudo ufw default deny incoming
$ sudo ufw default allow outgoing
$ sudo ufw allow OpenSSH
$ sudo ufw enable
$ sudo ufw status verbose
```

10.2 Reglas habituales

Necesidad	Regla
Web pública	`sudo ufw allow 'Nginx Full'` (80 y 443)
SSH solo desde la oficina	`sudo ufw allow from 203.0.113.10 to any port 22 proto tcp`
PostgreSQL solo desde la red interna	`sudo ufw allow from 10.0.0.0/24 to any port 5432`
Limitar intentos de conexión SSH	`sudo ufw limit OpenSSH`
Bloquear una IP	`sudo ufw insert 1 deny from 198.51.100.7`

10.3 Ver y borrar reglas

```
$ sudo ufw status numbered
$ sudo ufw delete 3
$ sudo ufw logging medium     # registra en /var/log/ufw.log
```

10.4 El caso Docker

Docker escribe sus propias reglas de red y puede publicar puertos saltándose UFW. Si usas contenedores, publica los puertos solo en localhost (`-p 127.0.0.1:8069:8069`) y expón el servicio a través de un proxy inverso (Nginx) protegido por UFW.

10.5 Cortafuegos en capas

Si tu servidor está en la nube (AWS, Hetzner, OVH, Azure), usa también el cortafuegos del proveedor. Así un error en UFW no deja el servidor expuesto: es la defensa en profundidad del capítulo 9 aplicada.

EJERCICIOS

1. Configura UFW para una máquina que sirva una web y solo acepte SSH desde tu IP.
2. Desde otra máquina, verifica con `nmap` que solo ves los puertos que has abierto.

CAPÍTULO 11

SSH seguro

SSH es la puerta principal de cualquier servidor Linux y la más atacada: un servidor recién publicado en Internet recibe intentos de acceso en minutos. Con claves en lugar de contraseñas, esos ataques dejan de funcionar.

11.1 Crear y usar claves

En tu ordenador (no en el servidor):

```
$ ssh-keygen -t ed25519 -C "marlon@portatil"
$ ssh-copy-id usuario@192.168.1.50
$ ssh usuario@192.168.1.50
```

Protege la clave privada con una frase de paso. La clave privada (`~/.ssh/id_ed25519`) nunca sale de tu equipo; la pública (`.pub`) es la que va al servidor.

11.2 Configuración endurecida

No edites el archivo principal: crea `/etc/ssh/sshd_config.d/99-hardening.conf`:

```
PermitRootLogin no
PasswordAuthentication no
KbdInteractiveAuthentication no
PubkeyAuthentication yes
AllowUsers marlon deploy
MaxAuthTries 3
LoginGraceTime 30
X11Forwarding no
ClientAliveInterval 300
ClientAliveCountMax 2
```

> **Atención:** antes de reiniciar, comprueba la sintaxis y mantén abierta una sesión de reserva.

```
$ sudo sshd -t && sudo systemctl reload ssh
```

Abre una **segunda** terminal y verifica que entras. Solo entonces cierra la primera.

11.3 Doble factor

Para servidores críticos, añade un código temporal (TOTP) con `libpam-google-authenticator`, o usa llaves físicas FIDO2 con claves del tipo `ed25519-sk`, que exigen tocar la llave en cada conexión.

11.4 ¿Cambiar el puerto 22?

Mover SSH a otro puerto reduce el ruido en los logs, pero no es una medida de seguridad real: un escaneo lo encuentra en segundos. Las claves, `AllowUsers`, Fail2ban y el filtrado por IP sí lo son.

11.5 Comodidad: el archivo config

En tu equipo, `~/.ssh/config` evita recordar IPs y usuarios:

```
Host produccion
    HostName 203.0.113.20
    User deploy
    IdentityFile ~/.ssh/id_ed25519
```

Ahora basta con `ssh produccion`.

EJERCICIOS

1. Desactiva el acceso por contraseña en tu servidor de laboratorio sin perder el acceso.
2. Busca los intentos fallidos con `grep 'Failed' /var/log/auth.log`.

CAPÍTULO
12

Fail2ban, AppArmor y auditoría

Tres herramientas cubren tres momentos del ataque: Fail2ban bloquea al que insiste, AppArmor encierra al que consigue entrar y auditd deja constancia de lo que hizo.

12.1 Fail2ban: bloquear a los insistentes

Fail2ban lee los logs y bloquea en el cortafuegos las IP que fallan demasiadas veces.

```
$ sudo apt install fail2ban
$ sudo nano /etc/fail2ban/jail.local
```

```
[DEFAULT]
bantime = 1h
findtime = 10m
maxretry = 5
bantime.increment = true
ignoreip = 127.0.0.1/8 203.0.113.10

[sshd]
enabled = true
```

```
$ sudo systemctl enable --now fail2ban
$ sudo fail2ban-client status sshd
$ sudo fail2ban-client set sshd unbanip 198.51.100.7
```

Incluye siempre tu IP de oficina en `ignoreip` para no bloquearte a ti mismo.

12.2 AppArmor: encerrar programas

AppArmor viene activo en Ubuntu. Asigna a cada programa un perfil que limita qué archivos y capacidades puede usar, aunque se ejecute como root.

```
$ sudo aa-status                          # perfiles cargados y su modo
$ sudo apt install apparmor-utils
$ sudo aa-enforce /etc/apparmor.d/usr.sbin.nginx
$ sudo aa-complain /etc/apparmor.d/usr.sbin.nginx   # solo registra
```

El modo *complain* sirve para probar un perfil nuevo sin romper la aplicación; cuando no aparezcan avisos en los logs, pásalo a *enforce*.

12.3 auditd: quién tocó qué

```
$ sudo apt install auditd
$ sudo auditctl -w /etc/passwd -p wa -k usuarios
$ sudo auditctl -w /etc/ssh/sshd_config -p wa -k ssh
$ sudo ausearch -k usuarios -i
```

Para que las reglas sobrevivan al reinicio, guárdalas en `/etc/audit/rules.d/hardening.rules`.

12.4 Integridad de archivos y rootkits

- **AIDE** toma una foto de los archivos del sistema y avisa si cambian: `sudo apt install aide && sudo aideinit`.
- **rkhunter** busca rootkits conocidos: `sudo rkhunter --check`.
- **ClamAV** analiza archivos subidos por usuarios o recibidos por correo.

EJERCICIOS

1. Instala Fail2ban, provoca cinco fallos de SSH desde otra máquina y comprueba el bloqueo.
2. Crea una regla de auditd sobre `/etc/sudoers` y verifica que registra un cambio.

CAPÍTULO
13

Copias de seguridad y cifrado

Una copia que no se ha probado a restaurar no es una copia. Frente al ransomware, la copia externa e inmutable es la última línea de defensa.

13.1 La regla 3-2-1

- **3** copias de los datos.
- En **2** soportes distintos.
- **1** fuera de la oficina (y, a ser posible, inmutable o desconectada).

13.2 Copias rápidas con rsync

```
$ rsync -aAXv --delete /home/ /mnt/backup/home/
$ rsync -az -e ssh /var/lib/miapp/ backup@nas:/copias/miapp/
```

13.3 Copias cifradas y versionadas con restic

restic cifra, deduplica y guarda versiones. Funciona contra un disco, un servidor SFTP o almacenamiento S3.

```
$ sudo apt install restic
$ restic -r sftp:backup@nas:/restic init
$ restic -r sftp:backup@nas:/restic backup /etc /home /var/lib/miapp
$ restic -r sftp:backup@nas:/restic snapshots
$ restic -r sftp:backup@nas:/restic forget --keep-daily 7 \
      --keep-weekly 4 --keep-monthly 12 --prune
$ restic -r sftp:backup@nas:/restic restore latest --target /tmp/prueba
```

13.4 Bases de datos

No copies los archivos de una base de datos en marcha. Vuelca primero:

```
$ sudo -u postgres pg_dump -Fc mibd > /var/backups/mibd_$(date +%F).dump
```

13.5 Cifrado de datos

Qué proteger	Herramienta
Disco completo	LUKS (en la instalación)
Un disco o USB adicional	`cryptsetup luksFormat /dev/sdb1`
Archivos sueltos	`gpg -c documento.pdf`
Tráfico web	Certificados TLS con Let's Encrypt (`certbot`)

Guarda la clave de recuperación de LUKS en un gestor de contraseñas o en papel en lugar seguro: sin ella, los datos son irrecuperables.

EJERCICIOS

1. Programa una copia diaria con restic usando un timer de systemd.
2. Restaura un archivo concreto desde una copia de hace dos días y mide cuánto tardas.

CAPÍTULO
14

Monitorización, logs y respuesta a incidentes

Detectar pronto convierte un desastre en un susto. Este capítulo enseña dónde mirar cada día y qué hacer, en orden, cuando sospechas una intrusión.

14.1 Dónde están los logs

Archivo o comando	Qué registra
`journalctl -u ssh --since today`	Todo lo que registra un servicio
`journalctl -p err -b`	Errores desde el último arranque
`/var/log/auth.log`	Inicios de sesión, `sudo`, SSH
`/var/log/syslog`	Eventos generales del sistema
`/var/log/ufw.log`	Conexiones bloqueadas por el cortafuegos
`/var/log/nginx/access.log`	Peticiones web

14.2 Revisión diaria de cinco minutos

```
$ last -n 20                                  # quién entró
$ sudo grep -c 'Failed password' /var/log/auth.log
$ sudo fail2ban-client status sshd
$ sudo ss -tulpn                              # ¿algún puerto nuevo?
$ df -h && free -h                            # disco y memoria
```

Para no depender de la memoria, instala `logwatch` y recibe un resumen diario por correo.

14.3 Señales de alarma

- Usuarios o claves SSH en `authorized_keys` que no reconoces.
- Procesos con nombres raros o CPU al 100 % sin motivo (minería de criptomonedas).

- Tareas nuevas en `crontab -l` o en `/etc/cron.d/`.
- Conexiones salientes hacia IP desconocidas en `ss -tp`.
- Logs borrados o con huecos de tiempo.

14.4 Respuesta a un incidente, paso a paso

1. **Mantén la calma y documenta**: hora, qué viste, qué comandos ejecutas.
2. **Aísla**: desconecta la máquina de la red o restringe el tráfico en el cortafuegos del proveedor. No la apagues todavía: perderías la memoria.
3. **Preserva evidencias**: copia logs y, si puedes, toma una instantánea del disco.
4. **Evalúa el alcance**: qué cuentas, datos y máquinas están afectados.
5. **Erradica**: reinstala desde cero; un sistema comprometido no se limpia, se reemplaza.
6. **Recupera**: restaura datos desde una copia anterior al incidente y cambia todas las credenciales.
7. **Notifica**: si hay datos personales afectados, el RGPD exige comunicarlo a la autoridad de control (en España, la AEPD) en un máximo de 72 horas.
8. **Aprende**: escribe qué falló y qué medida lo habría evitado.

EJERCICIOS

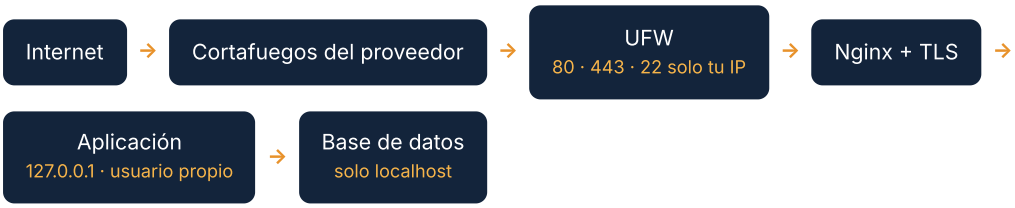
1. Crea un script `revision.sh` con los comandos de la revisión diaria.
2. Simula un incidente en tu laboratorio y recorre los ocho pasos por escrito.

CAPÍTULO
15

Proyecto final: un servidor de producción blindado

El proyecto final reúne todo el libro: montar desde cero un servidor Ubuntu LTS que publica una aplicación web detrás de Nginx con HTTPS, y superar la checklist de abajo. Al terminar, repite la auditoría de Lynis y compara con tu puntuación del capítulo 9.

15.1 Arquitectura



Además, una copia cifrada diaria sale del servidor hacia un almacenamiento externo.

15.2 Checklist de seguridad

SISTEMA

- ☐ Ubuntu LTS con soporte vigente
- ☐ `unattended-upgrades` activo (y Livepatch si hay Ubuntu Pro)
- ☐ Disco cifrado con LUKS cuando el hardware lo permita
- ☐ Ajustes de `sysctl` del capítulo 9 aplicados
- ☐ Servicios innecesarios desactivados

ACCESOS

- ☐ Root bloqueado y sin acceso por SSH
- ☐ SSH solo con claves; contraseñas desactivadas
- ☐ `AllowUsers` limitado a las cuentas necesarias
- ☐ Reglas de `sudo` mínimas y revisadas
- ☐ Cada aplicación con su usuario de servicio sin shell

RED

- ☐ UFW con política de denegar entradas
- ☐ Solo los puertos imprescindibles en `ss -tulpn`
- ☐ Bases de datos escuchando solo en localhost o red interna
- ☐ HTTPS con certificado válido y redirección desde HTTP

DETECCIÓN

- ☐ Fail2ban activo para SSH (y para la web si procede)
- ☐ AppArmor en modo *enforce* para los servicios expuestos
- ☐ auditd vigilando `/etc/passwd`, `/etc/sudoers` y la configuración de SSH
- ☐ Revisión diaria o resumen de logwatch por correo

RECUPERACIÓN

- ☐ Copia 3-2-1 cifrada y automática
- ☐ Restauración probada en el último mes
- ☐ Plan de respuesta a incidentes escrito y a mano

15.3 Mantenimiento recomendado

Frecuencia	Tarea
Diaria	Revisar logs y alertas
Semanal	Revisar actualizaciones pendientes y reinicios
Mensual	Probar una restauración; revisar usuarios y claves SSH
Trimestral	Auditoría con Lynis; revisar reglas de cortafuegos
Anual	Revisar el modelo de amenazas y planificar la migración de LTS

APÉNDICE

Apéndice. Chuleta de comandos

Área	Comando	Para qué
Sistema	`lsb_release -a` · `uname -r`	Versión de Ubuntu y del kernel
Sistema	`df -h` · `free -h` · `uptime`	Disco, memoria y carga
Archivos	`ls -la` · `cd` · `pwd`	Moverse y listar
Archivos	`find / -name X` · `grep -ri X ruta`	Buscar archivos y texto
Permisos	`chmod 640 f` · `chown u:g f`	Cambiar permisos y dueños
Usuarios	`adduser` · `usermod -aG` · `passwd -l`	Crear, agrupar, bloquear
Paquetes	`apt update && apt upgrade`	Actualizar el sistema
Servicios	`systemctl status/restart/enable X`	Gestionar servicios
Logs	`journalctl -u X -f` · `tail -f /var/log/auth.log`	Ver registros en vivo
Red	`ip a` · `ss -tulpn` · `netplan try`	Interfaces, puertos, configuración
Cortafuegos	`ufw status numbered` · `ufw allow X`	Ver y abrir puertos
SSH	`ssh-keygen -t ed25519` · `sshd -t`	Claves y validar configuración
Fail2ban	`fail2ban-client status sshd`	IP bloqueadas
AppArmor	`aa-status` · `aa-enforce perfil`	Estado y activar perfiles
Auditoría	`ausearch -k clave -i` · `lynis audit system`	Investigar y auditar
Copias	`restic backup` · `restic restore latest`	Copiar y restaurar

Tu servidor ya está siendo atacado.
La pregunta es si está preparado.

Cada día, miles de bots recorren Internet probando contraseñas y buscando sistemas sin actualizar. *Ubuntu Blindado* te enseña a dominar Linux desde cero y a convertir cualquier instalación de Ubuntu en un sistema robusto, vigilado y recuperable.

En sus quince capítulos aprenderás a:

- Manejar la terminal con soltura y entender permisos, usuarios y servicios.
- Configurar redes, cortafuegos UFW y accesos SSH solo con claves.
- Detectar y bloquear ataques con Fail2ban, AppArmor y auditd.
- Diseñar copias de seguridad cifradas que resisten al ransomware.
- Responder a un incidente paso a paso, incluido lo que exige el RGPD.

Con ejercicios en cada capítulo, un proyecto final y una checklist lista para aplicar en producción, es la guía práctica para técnicos, desarrolladores y consultores que despliegan aplicaciones en Linux.

Marlon Falcón Hernández es consultor especializado en ERP Odoo y fundador de Implemento BIM 2.0. Desde Valencia implanta y despliega sistemas de gestión empresarial sobre servidores Linux para clientes en España e Iberoamérica.

IMPLEMENTO BIM 2.0 · SERIE TÉCNICA