```
$ sudo harden --erp --linux
```

PROTECCIÓN **AVANZADA** de sistemas **ERP** sobre Linux

Bastionado, aislamiento, detección y recuperación para plataformas empresariales críticas

Marlon Falcón Hernández

Implemento BIM 2.0 · Primera edición · Valencia, 2026

Protección avanzada de sistemas ERP sobre Linux

Oct 31, 2026 · @generame una imagen de 3 arquitectos estilo pizaso

Portada

PROTECCIÓN AVANZADA DE SISTEMAS ERP SOBRE LINUX

Bastionado, aislamiento, detección y recuperación para plataformas empresariales críticas

Del servidor desnudo a un ERP en producción capaz de resistir intrusiones, ransomware y errores humanos. Válido para cualquier ERP que corra sobre Linux (Odoo, ERPNext, Dolibarr, Tryton o SAP), con ejemplos concretos sobre la pila más habitual: aplicación web y PostgreSQL.

Marlon Falcón Hernández

Primera edición · Valencia, 2026

Prólogo: el ERP es la empresa

Cuando un ERP cae, la empresa deja de facturar, de comprar y de pagar nóminas. Cuando un ERP se filtra, salen a la calle clientes, precios, márgenes y datos personales. Este libro existe porque la mayoría de las instalaciones que se ven en producción protegen el ERP como si fuera una web corporativa más.

La obra está pensada para administradores de sistemas, consultores ERP, desarrolladores que despliegan sus propios módulos y responsables de TI de pymes. Se asume soltura básica con la terminal de Linux; todo lo demás se construye capítulo a capítulo, con una idea central: **defensa en profundidad**. Ninguna capa es perfecta, así que cada una debe frenar lo que la anterior deje pasar.

Cómo leerlo

- Los capítulos 1 y 2 dan el modelo mental: qué se protege, de quién y con qué arquitectura.
- Los capítulos 3 a 10 bastionan cada capa, de abajo arriba: sistema, red, web, aplicación, base de datos, aislamiento, identidades y secretos.
- Los capítulos 11 a 13 asumen que algo fallará: copias, detección y respuesta.
- Los capítulos 14 y 15 cierran con el marco legal y un proyecto final con checklist.

Convenciones

- `$` indica un comando de usuario normal; `#` o `sudo`, uno con privilegios.
- Los ejemplos usan Debian 12/13 y Ubuntu 24.04 LTS; las diferencias con RHEL/Rocky se señalan cuando importan.
- Los ejemplos concretos usan Odoo con PostgreSQL por ser una pila muy extendida, pero cada medida se explica de forma genérica y vale para ERPNext, Dolibarr, Tryton, SAP u otro ERP sobre Linux. Rutas, puertos y nombres (`odoo`, `erp.ejemplo.com`, `10.10.20.0/24`) son ilustrativos: adáptalos a tu entorno.
- **Atención** marca acciones que pueden dejarte fuera del servidor o destruir datos. Pruébalas antes en un laboratorio.

Capítulo 1: El ERP como activo crítico y su modelo de amenazas

No se puede proteger bien lo que no se ha descrito. Antes de tocar un solo archivo de configuración hay que responder tres preguntas: qué activos contiene el ERP, quién querría atacarlos y por qué caminos podría llegar.

1.1 Qué hay dentro de un ERP

Activo	Ejemplo	Impacto si se pierde o filtra
Datos contables y fiscales	Asientos, facturas, IVA	Sanciones, imposibilidad de cerrar ejercicio
Datos personales	Clientes, empleados, nóminas	Brecha RGPD, multas, daño reputacional
Secretos comerciales	Tarifas, márgenes, proveedores	Pérdida de ventaja competitiva
Credenciales e integraciones	Claves de API, bancos, pasarelas	Fraude, movimientos no autorizados
Disponibilidad	El propio servicio	Paro operativo: no se vende ni se compra
Código propio	Módulos a medida	Robo de propiedad intelectual

1.2 Quién ataca

- **Bandas de ransomware**: buscan cifrar base de datos y copias para extorsionar. Es la amenaza más probable para una pyme.
- **Atacantes oportunistas**: escanean Internet buscando gestores de bases de datos expuestos, paneles con contraseñas por defecto y versiones sin parchear.

- **Fraude interno**: un empleado con más permisos de los necesarios que modifica cuentas bancarias de proveedores o borra rastros.
- **Cadena de suministro**: un módulo de terceros malicioso o vulnerable, una dependencia de Python comprometida, un partner con acceso que pierde sus credenciales.
- **Error humano**: un `DROP DATABASE` en el servidor equivocado, un puerto abierto "solo para probar".

1.3 Superficie de ataque típica

1. Interfaz web del ERP (formulario de inicio de sesión, gestor de bases de datos, APIs REST, SOAP o RPC).
2. Acceso administrativo (SSH, paneles de hosting, consolas del proveedor cloud).
3. Base de datos (puerto 5432 expuesto o accesible desde la red de oficina).
4. Integraciones salientes y entrantes (webhooks, conectores bancarios, correo).
5. Copias de seguridad (a menudo menos protegidas que el original).
6. Estaciones de trabajo de usuarios y administradores.

1.4 Método: STRIDE en diez minutos

Para cada componente pregunta: ¿se puede **S**uplantar una identidad? ¿**T**ampering, alterar datos? ¿**R**epudiar una acción por falta de registros? ¿Divulgar **I**nformación? ¿Denegar el servicio (**D**oS)? ¿**E**levar privilegios? Anota cada respuesta afirmativa como riesgo, con probabilidad e impacto de 1 a 5. Los riesgos con producto ≥ 12 se tratan primero.

Regla de oro: el objetivo no es la seguridad absoluta, sino que atacar tu ERP sea más caro que atacar al siguiente, y que si algo cae, vuelva en horas y no en semanas.

Ejercicios

1. Enumera los cinco activos más valiosos de un ERP que conozcas y quién tiene hoy acceso a cada uno.
2. Aplica STRIDE al formulario de inicio de sesión de tu ERP y prioriza los tres riesgos más altos.

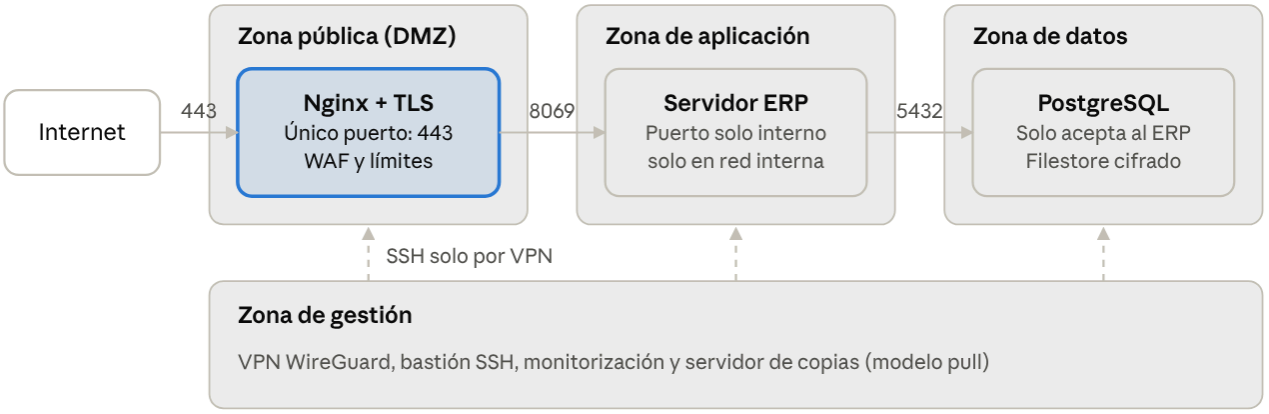
Capítulo 2: Arquitectura de referencia segura

La arquitectura decide el 70 % de la seguridad antes de escribir una sola regla. El principio es simple: **solo el proxy inverso habla con Internet; la base de datos no habla con nadie salvo con la aplicación.**

2.1 Las cuatro zonas

1. **Zona pública (DMZ)**: proxy inverso Nginx con TLS, WAF opcional. Único servicio con puertos 80/443 abiertos al exterior.
2. **Zona de aplicación**: servidores del ERP (Odoo, workers, cron). Escuchan solo en la red interna, nunca en `0.0.0.0` público.
3. **Zona de datos**: PostgreSQL y almacenamiento de adjuntos (filestore). Acepta conexiones solo desde la zona de aplicación.
4. **Zona de gestión**: bastión SSH o VPN WireGuard, monitorización, copias. Los administradores entran por aquí y solo por aquí.

Solo el proxy toca Internet; la base solo habla con el ERP



arquitectura por zonas · 4 zonas, 3 flujos permitidos

Cada flecha continua es el único flujo permitido entre zonas; la gestión entra en todas, pero solo a través de la VPN.

2.2 Tamaños de despliegue

Escenario	Usuarios	Topología recomendada
Pyme pequeña	Hasta 30	Un servidor, zonas separadas por cortafuegos local y sockets; base de datos en `localhost`
Pyme mediana	30–200	Dos servidores: aplicación + proxy, y base de datos dedicada en red privada
Empresa	200+	Proxy en alta disponibilidad, varios nodos de aplicación, PostgreSQL con réplica, red de gestión aislada

Incluso en un solo servidor la separación lógica se mantiene: el proxy escucha fuera, Odoo en `127.0.0.1:8069`, PostgreSQL en socket Unix.

2.3 Principios que guían todo el libro

- **Mínimo privilegio**: cada proceso, usuario y regla tiene exactamente lo que necesita.
- **Denegación por defecto**: se cierra todo y se abre lo justificado.
- **Separación de funciones**: quien despliega código no administra la base de datos en producción.
- **Inmutabilidad**: la configuración vive en un repositorio y se aplica con automatización (Ansible); los cambios a mano son la excepción documentada.
- **Asumir la brecha**: registros fuera del servidor, copias inmutables y un plan de respuesta ensayado.

2.4 Entornos

Producción, preproducción y desarrollo viven en redes y credenciales distintas. **Nunca** se copia una base de producción a desarrollo sin anonimizar datos personales: es una de las brechas RGPD más frecuentes.

Ejercicios

1. Dibuja la arquitectura actual de tu ERP y marca cada puerto abierto a Internet.
2. Decide qué escenario de la tabla 2.2 te corresponde y qué cambio te acercaría a él.

Capítulo 3: Bastionado del sistema operativo base

Un ERP seguro sobre un Linux descuidado es una caja fuerte con la puerta trasera abierta. Este capítulo deja el sistema base mínimo, actualizado y endurecido antes de instalar nada más.

3.1 Instalación mínima y particionado

Instala sin entorno gráfico ni servicios extra. Separa particiones para limitar daños y aplicar opciones de montaje restrictivas:

Punto de montaje	Opciones	Motivo
`/tmp`	`nodev,nosuid,noexec`	Impide ejecutar cargas útiles descargadas a temporales
`/var/tmp`	`nodev,nosuid,noexec`	Igual que `/tmp`

Punto de montaje	Opciones	Motivo
`/var/log`	`nodev,nosuid,noexec`	Evita que un disco lleno de logs tumbe el sistema
`/var/lib/postgresql`	`nodev,nosuid`	Datos de la base aislados
`/home`	`nodev,nosuid`	Usuarios sin binarios privilegiados

3.2 Actualizaciones automáticas de seguridad

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

Revisa `/etc/apt/apt.conf.d/50unattended-upgrades` para limitarlo al origen `-security` y programa los reinicios del kernel en ventana de mantenimiento. En RHEL/Rocky el equivalente es `dnf-automatic` con `upgrade_type = security`.

3.3 Reducir la superficie

```
$ systemctl list-unit-files --state=enabled
$ sudo ss -tulpn
$ sudo systemctl disable --now cups avahi-daemon bluetooth
$ sudo apt purge telnet rsh-client
```

Cada servicio en escucha debe tener un dueño y una razón. Si no la tiene, se desactiva.

3.4 Parámetros del kernel

Crea `/etc/sysctl.d/90-erp-hardening.conf`:

```
# Red
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.all.accept_source_route = 0
net.ipv4.tcp_syncookies = 1
net.ipv4.icmp_echo_ignore_broadcasts = 1
# Memoria y procesos
kernel.kptr_restrict = 2
kernel.dmesg_restrict = 1
kernel.yama.ptrace_scope = 1
kernel.unprivileged_bpf_disabled = 1
fs.protected_symlinks = 1
fs.protected_hardlinks = 1
fs.suid_dumpable = 0
```

Aplica con `sudo sysctl --system`.

3.5 Tiempo, registros y banner

- Sincroniza la hora con `chrony` o `systemd-timesyncd`: sin hora fiable, los logs no sirven como prueba.
- Haz persistente el journal (`Storage=persistent` en `/etc/systemd/journald.conf`).
- Añade un aviso legal en `/etc/issue.net`: refuerza la posición jurídica ante un acceso no autorizado.

3.6 Medir el resultado

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

Apunta el *hardening index* inicial y repite la auditoría tras cada capítulo. Para un baremo formal, aplica los perfiles CIS Benchmark de tu distribución con OpenSCAP.

Ejercicios

1. Ejecuta Lynis en un servidor de pruebas, corrige cinco avisos y compara el índice antes y después.
2. Lista los servicios en escucha y justifica por escrito cada uno.

Capítulo 4: Red, segmentación y cortafuegos con nftables

El cortafuegos traduce la arquitectura del capítulo 2 en reglas. Usamos `nftables`, el estándar actual del kernel; UFW y firewalld son capas sobre él y valen para casos simples, pero aquí queremos control fino.

4.1 Matriz de flujos

Antes de escribir reglas, escribe la matriz. Todo lo que no figure está prohibido.

Origen	Destino	Puerto	Motivo
Internet	Proxy	443/tcp (y 80 solo para redirigir)	Acceso de usuarios
Proxy	Aplicación	8069, 8072/tcp	HTTP y longpolling/websocket de Odoo
Aplicación	Base de datos	5432/tcp	Consultas SQL
VPN de gestión	Todos	22/tcp	Administración
Aplicación	Internet	443, 587/tcp	APIs, actualizaciones, correo saliente
Monitorización	Todos	9100/tcp	Métricas

4.2 Reglas base del servidor de aplicación

`/etc/nftables.conf`:

```
#!/usr/sbin/nft -f
flush ruleset

define PROXY = 10.10.10.5
define GESTION = 10.10.99.0/24

table inet filtro {
  chain entrada {
    type filter hook input priority 0; policy drop;
    ct state established,related accept
    ct state invalid drop
    iif lo accept
    ip saddr $GESTION tcp dport 22 ct state new limit rate 10/minute accept
    ip saddr $PROXY tcp dport { 8069, 8072 } accept
```

```
    ip saddr $GESTION tcp dport 9100 accept
    icmp type echo-request limit rate 5/second accept
    log prefix "nft-drop-in: " limit rate 5/minute drop
  }
  chain salida {
    type filter hook output priority 0; policy drop;
    ct state established,related accept
    oif lo accept
    ip daddr 10.10.30.10 tcp dport 5432 accept
    tcp dport { 443, 587 } accept
    udp dport { 53, 123 } accept
  }
  chain reenvio {
    type filter hook forward priority 0; policy drop;
  }
}
```

```
$ sudo nft -c -f /etc/nftables.conf   # comprobar sintaxis
$ sudo systemctl enable --now nftables
```

Atención: carga las reglas con una sesión SSH de respaldo abierta y un `at now + 5 minutes` que restaure las anteriores. Un `policy drop` mal escrito te deja fuera.

4.3 Filtrar también la salida

La política de salida en `drop` es la regla más olvidada y una de las más eficaces: un servidor comprometido no puede descargar herramientas ni exfiltrar datos a puertos arbitrarios. Si el ERP llama a pocas APIs, restringe por destino además de por puerto.

4.4 Acceso de gestión con WireGuard

SSH nunca debe estar abierto a Internet. Una VPN WireGuard en el bastión da un túnel cifrado, rápido y con claves por administrador:

```
$ sudo apt install wireguard
$ wg genkey | tee admin.key | wg pubkey > admin.pub
```

Cada administrador tiene su par de claves y su IP fija dentro de `10.10.99.0/24`. Dar de baja a alguien es borrar su `[Peer]`.

4.5 Protección frente a escaneos y DoS

- Limita conexiones nuevas por IP en el proxy (`limit rate`, conjuntos dinámicos de nftables).

- En la nube, usa además los grupos de seguridad del proveedor: dos cortafuegos independientes.
- Si el ERP es objetivo frecuente, pon delante un servicio anti-DDoS o CDN con WAF.

Ejercicios

1. Redacta la matriz de flujos de tu instalación y compárala con `ss -tulpn`.
2. Bloquea la salida a Internet salvo 443 y 587 y comprueba qué se rompe: eso es lo que no conocías.

Capítulo 5: Proxy inverso, TLS y protección web

El proxy inverso es la única cara pública del ERP. Cifra, filtra, limita y oculta. Usamos Nginx por ser el más extendido con Odoo; los conceptos valen para HAProxy, Caddy o Traefik.

5.1 Certificados

```
$ sudo apt install nginx certbot python3-certbot-nginx
$ sudo certbot --nginx -d erp.ejemplo.com
$ systemctl list-timers | grep certbot
```

La renovación es automática; comprueba que el temporizador existe y que alguien recibe los avisos de caducidad.

5.2 Configuración endurecida

```
limit_req_zone $binary_remote_addr zone=login:10m rate=5r/m;
limit_req_zone $binary_remote_addr zone=general:10m rate=20r/s;

upstream odoo     { server 10.10.20.10:8069; }
upstream odoochat { server 10.10.20.10:8072; }

server {
  listen 80;
  server_name erp.ejemplo.com;
  return 301 https://$host$request_uri;
}

server {
  listen 443 ssl;
  http2 on;
  server_name erp.ejemplo.com;

  ssl_protocols TLSv1.2 TLSv1.3;
```

```
  ssl_prefer_server_ciphers off;
  ssl_session_tickets off;
  ssl_stapling on;

  add_header Strict-Transport-Security "max-age=63072000; includeSubDomains"
always;
  add_header X-Content-Type-Options nosniff always;
  add_header X-Frame-Options SAMEORIGIN always;
  add_header Referrer-Policy strict-origin-when-cross-origin always;
  add_header Permissions-Policy "camera=(), microphone=(), geolocation=()"
always;

  server_tokens off;
  client_max_body_size 64m;
  proxy_read_timeout 720s;

  location ~ ^/web/database/ { deny all; }
  location = /web/login {
    limit_req zone=login burst=5 nodelay;
    proxy_pass http://odoo;
  }
  location /websocket {
    proxy_pass http://odoochat;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
  }
  location / {
    limit_req zone=general burst=40;
    proxy_set_header Host $host;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_redirect off;
    proxy_pass http://odoo;
  }
}
```

Las claves de esta configuración:

- **Bloquear `/web/database/`**: el gestor de bases de datos de Odoo permite crear, duplicar, descargar y borrar bases. Nunca debe ser accesible desde Internet.
- **Límite estricto en `/web/login`**: 5 intentos por minuto y por IP frenan la fuerza bruta antes de que llegue a la aplicación.

- **HSTS**: obliga al navegador a usar siempre HTTPS. Actívalo cuando estés seguro de que todo el dominio sirve HTTPS.
- **`server_tokens off`**: no regales la versión exacta al atacante.

En cualquier ERP, identifica las rutas de instalación, administración y gestión de bases de datos y ciérralas en el proxy. Por ejemplo, en Dolibarr el directorio `/install/` debe quedar bloqueado y protegido con su archivo `install.lock`.

5.3 Comprobar

```
$ sudo nginx -t && sudo systemctl reload nginx
$ curl -I https://erp.ejemplo.com
$ curl -o /dev/null -w "%{http_code}\n"
https://erp.ejemplo.com/web/database/manager
```

El último comando debe devolver `403`. Valida la calidad TLS con herramientas como `testssl.sh`.

5.4 WAF

Un cortafuegos de aplicación (ModSecurity con OWASP CRS, o el WAF de una CDN) añade detección de inyecciones y patrones de ataque. Empieza en modo detección: los ERP generan peticiones grandes y complejas, y un WAF mal ajustado rompe formularios legítimos.

5.5 Restringir por origen cuando se pueda

Si el ERP solo se usa desde oficinas o desde la VPN, la mejor protección es no publicarlo: `allow` con las IP de oficina y `deny all` para el resto. Los usuarios externos entran por VPN o por un proxy con identidad (Zero Trust).

Ejercicios

1. Comprueba si tu gestor de bases de datos responde desde Internet. Si lo hace, ciérralo hoy.
2. Simula 20 intentos de login seguidos con `curl` y verifica que Nginx devuelve `503` a partir del sexto.

Capítulo 6: Endurecer el servidor de aplicación ERP

La aplicación es donde viven la lógica de negocio y los permisos. Aquí se cometen los errores más caros: contraseña maestra débil, gestor de bases abierto, modo desarrollador en producción y módulos de origen dudoso.

6.0 Principios comunes a cualquier ERP

Todos los ERP web sobre Linux comparten los mismos puntos débiles, aunque cambien los nombres. Primero se aplican los principios; después, la sintaxis de tu producto. Los ejemplos de este capítulo usan Odoo.

Principio	Qué comprobar en tu ERP
Usuario de sistema propio y sin shell	El proceso no corre como `root` ni como un usuario humano
Código de solo lectura	Solo datos, adjuntos y logs son escribibles por el proceso
Funciones de administración ocultas	Instaladores, gestores de bases y paneles de configuración cerrados en producción
Contraseñas maestras robustas	Claves de instalación o superadministrador largas, únicas y fuera de archivos legibles
Escucha solo interna	El servidor de aplicación no se publica directamente; solo el proxy
Límites de recursos	Tiempo, memoria y número de procesos acotados
Modos de depuración desactivados	Ningún modo desarrollador, *debug* o trazas detalladas en producción
Extensiones revisadas	Módulos, *apps* o *add-ons* de terceros auditados antes de instalarse

6.1 Usuario de sistema dedicado

```
$ sudo adduser --system --group --home /opt/odoo --shell /usr/sbin/nologin
odoo
$ sudo chown -R root:odoo /opt/odoo/src
$ sudo chmod -R o-rwx /opt/odoo
$ sudo install -d -o odoo -g odoo -m 0750 /var/lib/odoo /var/log/odoo
```

El código pertenece a `root` y es solo lectura para `odoo`: si la aplicación es comprometida, no puede reescribirse a sí misma. Solo el filestore y los logs son escribibles.

6.2 Ejemplo con Odoo: `odoo.conf` de producción

```
[options]
admin_passwd = $pbkdf2-sha512$...   ; generado, nunca en claro
list_db = False
dbfilter = ^erp_prod$
proxy_mode = True
http_interface = 10.10.20.10
http_port = 8069
gevent_port = 8072
workers = 6
max_cron_threads = 2
limit_memory_soft = 2147483648
limit_memory_hard = 2684354560
limit_time_cpu = 600
limit_time_real = 1200
limit_request = 8192
db_host = 10.10.30.10
db_user = odoo_app
db_sslmode = verify-full
data_dir = /var/lib/odoo
logfile = /var/log/odoo/odoo.log
log_level = info
without_demo = all
server_wide_modules = base,web
```

```
$ sudo chown root:odoo /etc/odoo/odoo.conf
$ sudo chmod 640 /etc/odoo/odoo.conf
```

Los parámetros que más importan:

- **`list_db = False`** y **`dbfilter`**: ocultan el selector y el gestor de bases. Junto con el bloqueo del capítulo 5, son dos capas para la misma puerta.
- **`admin_passwd`**: la contraseña maestra permite borrar y restaurar bases. Usa 32+ caracteres aleatorios; Odoo la guarda con hash al cambiarla desde la interfaz.
- **`proxy_mode = True`**: necesario para que Odoo registre la IP real del cliente y no la del proxy.
- **`limit_*`** y **`workers`**: además de rendimiento, son protección contra agotamiento de recursos.
- **`http_interface`**: escucha solo en la red interna.

6.3 Seguridad dentro de la aplicación

- Desactiva el **modo desarrollador** para todos salvo administradores técnicos, y nunca lo dejes activo.
- Revisa **grupos y reglas de registro**: el usuario de almacén no necesita ver nóminas ni cuentas bancarias.
- Cambia el login `admin` por defecto y quítale uso diario: cada administrador con su cuenta nominal.
- Exige **2FA (TOTP)** a todos los usuarios internos (capítulo 9).
- Limita las **claves de API** a usuarios técnicos con permisos mínimos y caducidad.
- Activa la **política de contraseñas** (longitud mínima, comprobación de contraseñas filtradas).

6.4 Cadena de suministro: módulos y dependencias

1. Instala módulos solo de fuentes conocidas y revisa el código de los de terceros antes de llevarlos a producción: busca `sudo()`, `eval`, `safe_eval` con entradas de usuario, SQL construido con concatenación y controladores con `auth='public'` o `csrf=False`.
2. Fija versiones de dependencias Python en un entorno virtual y audítalas con `pip-audit`.
3. Despliega desde Git con etiquetas firmadas; nunca editando archivos en el servidor.
4. Sigue los avisos de seguridad del fabricante del ERP y aplica los parches en días, no meses.

Ejercicios

1. Busca en tus módulos propios todos los controladores con `auth='public'` y justifica cada uno.
2. Comprueba con `ls -l` que el usuario `odoo` no puede escribir en su propio código.

Capítulo 7: PostgreSQL blindado

La base de datos es el tesoro. Un atacante que llega a ella con permisos de superusuario no necesita nada más: puede leer todo, alterar asientos y, en muchos casos, ejecutar comandos en el sistema operativo.

7.1 Escuchar solo donde hace falta

`postgresql.conf`:

```
listen_addresses = '10.10.30.10'     # o 'localhost' en servidor único
password_encryption = scram-sha-256
ssl = on
ssl_cert_file = '/etc/postgresql/tls/server.crt'
ssl_key_file  = '/etc/postgresql/tls/server.key'
ssl_min_protocol_version = 'TLSv1.2'
log_connections = on
log_disconnections = on
log_line_prefix = '%m [%p] %q%u@%d %r '
log_min_duration_statement = 2000
shared_preload_libraries = 'pgaudit'
```

7.2 `pg_hba.conf`: quién entra y cómo

```
# TIPO    BASE      USUARIO    ORIGEN            MÉTODO
local     all       postgres                     peer
hostssl   erp_prod  odoo_app   10.10.20.10/32    scram-sha-256
hostssl   erp_prod  odoo_ro    10.10.99.0/24     scram-sha-256
host      all       all        0.0.0.0/0         reject
host      all       all        ::/0              reject
```

Sin `trust`, sin `md5` y con una regla final de rechazo explícita. El superusuario `postgres` solo entra por socket local.

7.3 Roles con mínimo privilegio

El error clásico es que la aplicación se conecte como superusuario. La aplicación ERP suele necesitar ser **propietario** de su base para crear tablas al instalar módulos, pero **no** necesita ser superusuario ni poder crear bases nuevas:

```
CREATE ROLE odoo_app LOGIN PASSWORD '...' NOSUPERUSER NOCREATEDB
NOCREATEROLE;
CREATE DATABASE erp_prod OWNER odoo_app;
REVOKE ALL ON DATABASE erp_prod FROM PUBLIC;

-- Rol de solo lectura para informes y BI
CREATE ROLE odoo_ro LOGIN PASSWORD '...' NOSUPERUSER;
GRANT CONNECT ON DATABASE erp_prod TO odoo_ro;
\c erp_prod
GRANT USAGE ON SCHEMA public TO odoo_ro;
GRANT SELECT ON ALL TABLES IN SCHEMA public TO odoo_ro;
ALTER DEFAULT PRIVILEGES FOR ROLE odoo_app IN SCHEMA public GRANT SELECT ON
TABLES TO odoo_ro;
```

Para herramientas de BI, mejor aún: expón vistas que excluyan columnas sensibles (IBAN, nóminas, documentos de identidad) y da acceso solo a ellas.

7.4 Auditoría con pgaudit

```
pgaudit.log = 'ddl, role, write'
pgaudit.log_relation = on
```

Registra cambios de estructura, de permisos y escrituras. En un ERP con mucho tráfico, `write` genera volumen: envía los registros fuera del servidor (capítulo 12) y ajusta la retención.

7.5 Otras medidas

- Rota la contraseña de `odoo_app` y guárdala en un gestor de secretos (capítulo 10).
- Limita conexiones: `max_connections` acorde a `workers` y, si hace falta, PgBouncer delante.
- Mantén PostgreSQL en una versión mayor con soporte y aplica las menores sin demora.
- Revisa periódicamente quién tiene qué: `\du` y `\dp` en `psql`.

Ejercicios

1. Comprueba con `SELECT rolname, rolsuper FROM pg_roles;` si tu aplicación usa un rol superusuario.
2. Intenta conectar a PostgreSQL desde una máquina no autorizada y confirma que `pg_hba.conf` la rechaza.

Capítulo 8: Aislamiento con systemd, AppArmor/SELinux y contenedores

Aunque se cuele una vulnerabilidad en la aplicación, el atacante debe encontrarse en una celda: sin poder leer el resto del sistema, sin escalar privilegios y sin abrir conexiones nuevas.

8.1 systemd como sandbox

`/etc/systemd/system/odoo.service`:

```
[Unit]
Description=Odoo ERP
After=network-online.target

[Service]
User=odoo
Group=odoo
ExecStart=/opt/odoo/venv/bin/python /opt/odoo/src/odoo-bin -c
/etc/odoo/odoo.conf
Restart=on-failure

NoNewPrivileges=yes
PrivateTmp=yes
PrivateDevices=yes
ProtectSystem=strict
ProtectHome=yes
ReadWritePaths=/var/lib/odoo /var/log/odoo
ProtectKernelTunables=yes
ProtectKernelModules=yes
ProtectKernelLogs=yes
ProtectControlGroups=yes
ProtectClock=yes
ProtectHostname=yes
RestrictSUIDSGID=yes
RestrictRealtime=yes
RestrictNamespaces=yes
LockPersonality=yes
RestrictAddressFamilies=AF_INET AF_INET6 AF_UNIX
SystemCallArchitectures=native
SystemCallFilter=@system-service
SystemCallFilter=~@privileged @resources
CapabilityBoundingSet=
UMask=0027
LimitNOFILE=65536

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

```
$ sudo systemctl daemon-reload && sudo systemctl restart odoo
$ systemd-analyze security odoo
```

El objetivo es bajar la puntuación de exposición de `systemd-analyze` por debajo de 3. **Atención**: no uses `MemoryDenyWriteExecute=yes` sin probar; algunas librerías Python lo incumplen. Si generas PDF con wkhtmltopdf, prueba la impresión de informes tras cada cambio.

8.2 AppArmor y SELinux

El control de acceso obligatorio limita qué archivos y capacidades puede usar un proceso aunque el usuario tenga permisos.

- **Debian/Ubuntu (AppArmor)**: genera un perfil para Odoo con `aa-genprof`, ejecútalo en modo *complain* durante una semana de uso real, ajústalo con `aa-logprof` y pásalo a *enforce*.
- **RHEL/Rocky (SELinux)**: mantén `enforcing`. Etiqueta rutas propias con `semanage fcontext` y `restorecon`, y permite la conexión del proxy con `setsebool -P httpd_can_network_connect 1`. Nunca resuelvas un problema poniendo SELinux en `permissive` de forma permanente.

8.3 Contenedores

Docker o Podman facilitan despliegues reproducibles, pero no son una barrera de seguridad por sí mismos.

Práctica	Cómo
Sin root	Podman rootless o `user: 101:101` en compose
Sistema de archivos de solo lectura	`read_only: true` con volúmenes solo para filestore y logs
Sin capacidades	`cap_drop: [ALL]`, `security_opt: [no-new-privileges:true]`
Imagen mínima y verificada	Imagen oficial fijada por digest, escaneada con Trivy
Redes separadas	Red interna sin salida para la base de datos
Sin el socket de Docker	Nunca montar `/var/run/docker.sock` en un contenedor
Límites	`mem_limit`, `pids_limit`, `cpus`

Ejercicios

1. Ejecuta `systemd-analyze security odoo` antes y después de aplicar la unidad del apartado 8.1 y anota la mejora.
2. Si usas contenedores, escanea tus imágenes con `trivy image` y corrige las vulnerabilidades críticas.

Capítulo 9: Identidad, accesos y MFA

La mayoría de las intrusiones no rompen nada: entran con credenciales robadas. Por eso cada acceso, humano o técnico, debe ser nominal, mínimo, con segundo factor y revocable en minutos.

9.1 SSH endurecido

`/etc/ssh/sshd_config.d/10-erp.conf`:

```
PermitRootLogin no
PasswordAuthentication no
KbdInteractiveAuthentication no
PubkeyAuthentication yes
AuthenticationMethods publickey
AllowGroups sshadmins
MaxAuthTries 3
LoginGraceTime 20
ClientAliveInterval 300
ClientAliveCountMax 2
X11Forwarding no
AllowAgentForwarding no
AllowTcpForwarding no
ListenAddress 10.10.99.1
```

- Claves **Ed25519**, preferiblemente en llave de seguridad FIDO2 (`ssh-keygen -t ed25519-sk`).
- Para equipos de cierto tamaño, certificados SSH firmados por una CA interna con caducidad de horas.
- Comprueba la configuración con `sudo sshd -t` antes de reiniciar y con una sesión abierta de respaldo.

9.2 sudo con trazabilidad

```
# /etc/sudoers.d/erp (editar con visudo)
Defaults use_pty, logfile="/var/log/sudo.log", log_input, log_output
%sshadmins ALL=(ALL) ALL
%erpdeploy ALL=(odoo) NOPASSWD: /usr/bin/systemctl restart odoo
```

El grupo de despliegue solo puede reiniciar el servicio, no tener una shell de root.

9.3 MFA en el ERP

Activa la autenticación en dos pasos (TOTP) para todos los usuarios internos y hazla **obligatoria** para administradores, contabilidad, RR. HH. y tesorería. Si la organización ya tiene proveedor de identidad (Entra ID, Google Workspace, Keycloak), integra el ERP por OAuth2/OIDC o SAML: el alta y la baja se gestionan en un único sitio y el MFA lo impone el proveedor.

9.4 Ciclo de vida de los accesos

Momento	Acción
Alta	Cuenta nominal, rol mínimo según puesto, MFA activado el primer día
Cambio de puesto	Se retiran los permisos anteriores antes de añadir los nuevos
Baja	Desactivar ERP, VPN y SSH el mismo día; rotar secretos compartidos que conociera
Trimestral	Revisión de usuarios activos, administradores y claves de API
Proveedores y partners	Acceso temporal, nominal, con caducidad y registro de sesión

9.5 Cuentas de servicio y API

Cada integración tiene su propio usuario técnico, con permisos solo sobre los modelos que usa, claves de API rotadas y, si es posible, restricción por IP en el proxy. Una integración que usa la cuenta de un administrador es una puerta trasera documentada.

Ejercicios

1. Lista los usuarios con derechos de administración en tu ERP. ¿Todos los necesitan?
2. Desactiva la autenticación por contraseña en SSH en un servidor de pruebas sin perder el acceso.

Capítulo 10: Secretos y cifrado

Un ERP maneja decenas de secretos: contraseña maestra, credenciales de base de datos, claves de API de bancos y pasarelas, certificados de facturación electrónica. Y almacena datos que deben cifrarse en tránsito, en reposo y en las copias.

10.1 Inventario de secretos

Secreto	Dónde suele estar	Dónde debería estar
Contraseña de PostgreSQL	`odoo.conf` en claro, legible por todos	Gestor de secretos o archivo `640 root:odoo`
`admin_passwd`	Contraseña corta y compartida	Hash, 32+ caracteres, en el gestor del equipo
Claves de API de terceros	Parámetros del sistema del ERP	Campos protegidos y acceso restringido a administradores
Certificado digital (SII, Verifactu, FacturaE)	Carpeta compartida	Almacén protegido, con contraseña y renovación controlada
Claves SSH y de copias	Portátiles de administradores	Llaves FIDO2 y gestor con auditoría

10.2 Gestores de secretos

Para entornos pequeños, un gestor de contraseñas de equipo con auditoría (Bitwarden/Vaultwarden, 1Password) es suficiente. Para entornos con automatización, HashiCorp Vault u OpenBao permiten credenciales dinámicas de PostgreSQL que caducan solas. Con Ansible, cifra las variables sensibles con `ansible-vault` y nunca subas secretos a Git: añade `gitleaks` como comprobación previa al commit.

10.3 Cifrado en tránsito

- Navegador → proxy: TLS 1.2/1.3 (capítulo 5).
- Proxy → aplicación: en un único servidor basta la interfaz local; entre servidores, red privada y, si cruza redes no confiables, TLS o WireGuard.
- Aplicación → base de datos: `sslmode=verify-full` (capítulo 7).
- Copias → destino remoto: siempre cifradas en origen.

10.4 Cifrado en reposo

```
# Volumen de datos cifrado con LUKS2
$ sudo cryptsetup luksFormat --type luks2 /dev/vdb
$ sudo cryptsetup open /dev/vdb datos_erp
$ sudo mkfs.ext4 /dev/mapper/datos_erp
```

LUKS protege frente al robo del disco o de una instantánea del proveedor, no frente a un atacante con sesión en el servidor encendido. En servidores remotos, desbloquea al arrancar con `clevis` + Tang o con TPM2 (`systemd-cryptenroll`) para no depender de teclear la frase.

El **filestore** de Odoo (adjuntos, facturas en PDF, documentos de empleados) suele contener tantos datos personales como la base: debe estar en el volumen cifrado y en las copias.

Ejercicios

1. Busca contraseñas en claro con `sudo grep -RIn "password" /etc /opt --include=*.conf` y mueve las que encuentres.
2. Pasa `gitleaks detect` sobre el repositorio de tus módulos y revisa el historial completo.

Capítulo 11: Copias inmutables y recuperación ante ransomware

Una copia que el atacante puede borrar no es una copia. El ransomware moderno busca primero los backups, los destruye y solo después cifra producción. Este capítulo diseña copias que sobreviven a un administrador comprometido.

11.1 Regla 3-2-1-1-0

- **3** copias de los datos (producción + 2).
- **2** soportes o servicios distintos.
- **1** copia fuera del centro de datos principal.
- **1** copia inmutable o desconectada.
- **0** errores en las pruebas de restauración.

11.2 Qué copiar

Base de datos **y** filestore, coherentes entre sí y del mismo momento. También `odoo.conf`, el código de módulos propios (en Git), configuración de Nginx, PostgreSQL y nftables, y las claves necesarias para descifrar las copias, guardadas aparte.

11.3 PostgreSQL: lógica y física

```
# Copia lógica diaria, formato comprimido
$ sudo -u postgres pg_dump -Fc -d erp_prod -f /backup/erp_prod_$(date
+%F).dump
```

Para recuperar a un punto concreto en el tiempo (PITR), usa copias físicas con archivado de WAL mediante **pgBackRest** o **Barman**. Con PITR puedes volver al minuto anterior a un borrado masivo o al inicio de un cifrado.

11.4 Filestore y configuración con restic

```
$ export RESTIC_REPOSITORY=s3:https://s3.ejemplo-proveedor.com/erp-backup
$ restic backup /var/lib/odoo /etc/odoo /etc/nginx /backup --tag diario
$ restic forget --keep-daily 14 --keep-weekly 8 --keep-monthly 12 --prune
$ restic check --read-data-subset=5%
```

restic cifra en origen; la clave del repositorio no debe vivir solo en el servidor que se copia.

11.5 Inmutabilidad real

1. **Object Lock** en almacenamiento S3 compatible, en modo *compliance*, con retención mínima de 30 días: nadie, ni siquiera el propietario, puede borrar antes.
2. **Modelo pull**: un servidor de copias aislado se conecta a producción y extrae los datos; producción no tiene credenciales para escribir ni borrar en el destino.
3. Credenciales del servidor de producción con permiso de **solo añadir** (por ejemplo, `rest-server --append-only` para restic).
4. Una copia mensual desconectada o en otra cuenta de nube con otro administrador.

11.6 Objetivos y pruebas

Define con la dirección dos cifras: **RPO** (cuántos datos se pueden perder, por ejemplo 15 minutos) y **RTO** (cuánto puede estar parado, por ejemplo 4 horas). Después pruébalas: una restauración completa en un servidor limpio cada trimestre, cronometrada y documentada. Comprueba que el ERP arranca, que los adjuntos se abren y que el último asiento coincide.

Atención: al restaurar en otro entorno, desactiva correo saliente, crons e integraciones antes de arrancar, o el ERP restaurado enviará facturas y pagos duplicados.

Ejercicios

1. Intenta borrar tu copia más reciente usando solo las credenciales que hay en el servidor de producción. Si puedes, el diseño falla.
2. Restaura la base y el filestore en un servidor nuevo y mide cuánto tardas.

Capítulo 12: Detección, auditoría y monitorización

Las defensas fallan; lo que marca la diferencia es enterarse en minutos y no en meses. Este capítulo monta los ojos del sistema: registros centralizados, integridad de archivos, bloqueo automático y alertas útiles.

12.1 Registros fuera del servidor

Un atacante con root borra los logs locales. Envía journal, Nginx, Odoo, PostgreSQL, sudo y auditd a un servidor central (rsyslog con TLS, Vector o el agente de un SIEM) en el que producción solo puede escribir.

12.2 auditd: quién tocó qué

`/etc/audit/rules.d/erp.rules`:

```
-w /etc/odoo/ -p wa -k erp_config
-w /opt/odoo/src/ -p wa -k erp_code
-w /etc/postgresql/ -p wa -k pg_config
-w /etc/nginx/ -p wa -k web_config
-w /etc/ssh/sshd_config -p wa -k ssh_config
-w /etc/sudoers -p wa -k privilegios
-w /etc/sudoers.d/ -p wa -k privilegios
-a always,exit -F arch=b64 -S execve -F euid=0 -F auid>=1000 -F auid!=unset -
k root_cmd
```

```
$ sudo augenrules --load
$ sudo ausearch -k erp_config -i
```

12.3 Integridad de archivos

AIDE crea una huella de binarios y configuraciones y avisa de cualquier cambio:

```
$ sudo apt install aide
$ sudo aideinit
$ sudo aide --check
```

Actualiza la base de referencia tras cada despliegue legítimo; un cambio sin despliegue asociado es una alerta.

12.4 Fail2ban para el ERP (ejemplo con Odoo)

Filtro `/etc/fail2ban/filter.d/odoo-login.conf`:

```
[Definition]
failregex = ^.* \d+ INFO \S+ odoo\.addons\.base\.models\.res_users: Login
failed for db:\S+ login:\S+ from <HOST>
ignoreregex =
```

Cárcel en `/etc/fail2ban/jail.d/odoo.local`:

```
[odoo-login]
enabled  = true
port     = http,https
filter   = odoo-login
logpath  = /var/log/odoo/odoo.log
maxretry = 5
findtime = 600
bantime  = 3600
banaction = nftables-multiport
```

Requiere `proxy_mode = True` para que el log muestre la IP real. Si el bloqueo debe aplicarse en el proxy y no en el servidor de aplicación, ejecuta Fail2ban allí leyendo los logs centralizados. Prueba el filtro con `fail2ban-regex`, ya que el formato del log puede variar entre versiones.

12.5 Qué alertar

Señal	Por qué importa
Inicio de sesión de administrador desde IP o país nuevo	Credenciales robadas
Picos de logins fallidos	Fuerza bruta o *credential stuffing*
Cambio en cuentas bancarias de proveedores	Fraude del CEO / proveedor
Exportaciones masivas desde el ERP	Exfiltración
Cambios en `odoo.conf`, código o reglas sudo fuera de ventana	Persistencia del atacante
Copia de seguridad fallida o más pequeña de lo normal	Sabotaje previo al ransomware
Disco, memoria o conexiones a PostgreSQL al límite	Degradación o DoS

Una plataforma como **Wazuh** reúne la mayoría de estas piezas (agentes, integridad, reglas y panel). Para métricas de rendimiento, Prometheus con `node_exporter` y `postgres_exporter`, y Grafana para paneles y alertas.

Ejercicios

1. Provoca cinco logins fallidos desde una IP de pruebas y comprueba que Fail2ban la bloquea.
2. Modifica un comentario en `odoo.conf` y localiza el evento con `ausearch`.

Capítulo 13: Respuesta a incidentes

El peor momento para improvisar es durante un incidente. Un plan breve, impreso y ensayado vale más que un manual de cien páginas que nadie ha leído.

13.1 Las seis fases

1. **Preparación**: contactos, accesos de emergencia, copias verificadas, este plan.
2. **Detección y análisis**: confirmar que es un incidente, medir alcance y gravedad.
3. **Contención**: frenar el daño sin destruir pruebas.
4. **Erradicación**: eliminar la causa (credencial, vulnerabilidad, persistencia).
5. **Recuperación**: volver a operar desde un estado limpio y vigilado.
6. **Lecciones aprendidas**: informe, medidas y actualización del plan en dos semanas.

13.2 Primeros 60 minutos ante sospecha de intrusión

- [ ] Abrir un registro del incidente con hora de cada acción (quién, qué, cuándo).
- [ ] **No apagar** el servidor: la memoria contiene pruebas. Aislarlo de la red (cortafuegos del proveedor o regla nftables que solo permita la IP del analista).
- [ ] Tomar instantánea del disco y, si es posible, volcado de memoria (por ejemplo, con AVML o LiME).
- [ ] Recoger evidencias volátiles: `ss -tupan`, `ps auxf`, `last -Fwx`, `who`, `lsof -nP`, crontabs y unidades systemd recientes.
- [ ] Copiar logs centralizados del periodo afectado a un almacén de solo lectura.
- [ ] Revocar credenciales potencialmente expuestas: claves SSH, contraseñas de administración, claves de API, `admin_passwd`, contraseña de PostgreSQL.
- [ ] Avisar a dirección y al responsable de protección de datos.

13.3 Ransomware: decisión rápida

Si hay cifrado en curso, la prioridad es cortar: aislar red y suspender copias automáticas para que no sobrescriban versiones buenas. La recuperación se hace **en servidores nuevos**, con sistema reinstalado desde cero, código desde Git y datos desde la última

copia inmutable anterior a la intrusión (no a la detección: los atacantes suelen llevar semanas dentro). Pagar no garantiza recuperar datos y puede tener implicaciones legales; consúltalo con asesoría jurídica y con las autoridades.

13.4 Obligaciones de notificación

- **RGPD**: si hay brecha de datos personales con riesgo para las personas, notificar a la autoridad de control (en España, la AEPD) en un máximo de **72 horas** desde que se tiene constancia, y a los afectados si el riesgo es alto.
- **NIS2**: las entidades incluidas deben enviar alerta temprana en **24 horas**, notificación en **72 horas** e informe final en un mes al CSIRT de referencia (en España, INCIBE-CERT o CCN-CERT según el caso).
- Denuncia ante las Fuerzas y Cuerpos de Seguridad del Estado cuando haya delito.

Confirma los plazos vigentes y tu encaje concreto con un asesor legal: la normativa de transposición puede variar.

13.5 Contactos que deben estar escritos antes

Rol	Responsabilidad
Coordinador del incidente	Decide y comunica; no ejecuta comandos
Técnico de sistemas	Contención y evidencias
Responsable del ERP / partner	Integridad funcional y datos
Delegado de protección de datos	Evaluación y notificación RGPD
Asesoría jurídica	Denuncias, contratos, comunicación externa
Proveedor cloud y de copias	Acceso de emergencia y restauración

Ejercicios

1. Redacta tu plan de una página con los contactos de la tabla 13.5.
2. Organiza un simulacro de mesa: "El lunes a las 8:00 el ERP muestra una nota de rescate". Cronometra cada decisión.

Capítulo 14: Cumplimiento normativo (RGPD, ENS, NIS2, ISO 27001)

Casi todo lo construido en este libro tiene, además de valor técnico, valor de cumplimiento. Este capítulo lo traduce al lenguaje de auditores y abogados. No sustituye el asesoramiento legal, pero te permite llegar a esa conversación con los deberes hechos.

14.1 Mapa de normas

Marco	A quién aplica	Qué exige en la práctica
RGPD y LOPDGDD	Toda organización que trate datos personales en la UE	Medidas técnicas adecuadas al riesgo, registro de actividades, notificación de brechas, privacidad desde el diseño
ENS (RD 311/2022)	Sector público español y sus proveedores tecnológicos	Medidas por categoría (básica, media, alta), auditoría y declaración o certificación de conformidad
NIS2	Entidades esenciales e importantes de sectores definidos, y su cadena de suministro	Gestión de riesgos, seguridad de proveedores, notificación de incidentes, responsabilidad de la dirección
ISO/IEC 27001:2022	Voluntaria; muy pedida en contratos B2B	Sistema de gestión, análisis de riesgos y los 93 controles del Anexo A

Si das servicio a la administración o a grandes empresas, es probable que te pidan ENS o ISO 27001 aunque la ley no te obligue directamente.

14.2 Del capítulo al control

Medida del libro	Capítulos	RGPD	ISO 27001 (Anexo A)
Modelo de amenazas y riesgos	1	Art. 32, 35	5.7, cláusula 6.1
Segmentación y cortafuegos	2, 4	Art. 32	8.20, 8.22
Bastionado y parches	3, 6	Art. 32	8.8, 8.9
TLS y cifrado en reposo	5, 10	Art. 32.1.a	8.24
Mínimo privilegio y MFA	7, 9	Art. 25, 32	5.15, 5.18, 8.2, 8.5

Medida del libro	Capítulos	RGPD	ISO 27001 (Anexo A)
Copias y pruebas de restauración	11	Art. 32.1.c	8.13
Registros y monitorización	12	Art. 32.1.d	8.15, 8.16
Respuesta y notificación	13	Art. 33, 34	5.24–5.28
Seguridad de proveedores y módulos	6	Art. 28	5.19–5.21

14.3 Evidencias que conviene guardar

- Análisis de riesgos firmado y revisado cada año.
- Informes de Lynis/OpenSCAP y de `systemd-analyze security` con fecha.
- Actas de revisión trimestral de accesos.
- Registros de pruebas de restauración con tiempos medidos.
- Registro de incidentes, aunque sean menores.
- Contratos de encargado del tratamiento con hosting, partners y proveedores de copias.

14.4 Datos personales dentro del ERP

Configura la retención: borra o anonimiza candidatos, leads y contactos inactivos cuando termine su base legal. Los datos fiscales, en cambio, deben conservarse durante los plazos que fija la normativa tributaria y mercantil. Documenta qué se guarda, cuánto tiempo y por qué.

Ejercicios

1. Rellena la tabla 14.2 con el estado real de tu instalación: hecho, parcial o pendiente.
2. Determina si tu organización o tus clientes entran en el ámbito de NIS2 o del ENS.

Capítulo 15: Proyecto final y checklist de bastionado ERP

El proyecto final consiste en desplegar desde cero un ERP en producción aplicando todo el libro, y verificarlo con la checklist. Hazlo primero en laboratorio (tres máquinas virtuales: proxy, aplicación y base de datos, más una de gestión) y después en real.

15.1 Orden de construcción

1. Modelo de amenazas y matriz de flujos por escrito (cap. 1, 4).
2. Sistemas base bastionados con Ansible, índice Lynis registrado (cap. 3).

3. VPN de gestión y SSH solo por VPN, sin contraseñas (cap. 4, 9).
4. nftables con entrada y salida en `drop` en las cuatro máquinas (cap. 4).
5. PostgreSQL con TLS, `pg_hba` cerrado, roles mínimos y pgaudit (cap. 7).
6. Odoo con usuario dedicado, `odoo.conf` de producción y unidad systemd aislada (cap. 6, 8).
7. Nginx con TLS, cabeceras, límites y gestor de bases bloqueado (cap. 5).
8. MFA para todos, revisión de grupos y claves de API (cap. 9).
9. Secretos fuera de archivos legibles, volúmenes cifrados (cap. 10).
10. Copias pull + inmutables y primera restauración cronometrada (cap. 11).
11. Logs centralizados, auditd, AIDE, Fail2ban y alertas (cap. 12).
12. Plan de respuesta impreso y simulacro (cap. 13).

15.2 Checklist de bastionado

Sistema

- [ ] Instalación mínima, particiones con `noexec` donde corresponde
- [ ] Actualizaciones de seguridad automáticas y ventana de reinicio definida
- [ ] `sysctl` endurecido y servicios innecesarios desactivados
- [ ] Hora sincronizada y journal persistente

Red y web

- [ ] Solo 443 (y 80 de redirección) expuestos a Internet
- [ ] SSH accesible solo desde la VPN de gestión
- [ ] Política de salida restrictiva
- [ ] TLS 1.2+ con HSTS y cabeceras de seguridad
- [ ] `/web/database/` devuelve 403 desde Internet
- [ ] Límite de peticiones en `/web/login`

Aplicación y datos

- [ ] `list_db = False`, `dbfilter` y `admin_passwd` robusta
- [ ] Código de solo lectura para el usuario `odoo`
- [ ] Modo desarrollador desactivado en producción
- [ ] PostgreSQL sin superusuario para la aplicación y con `scram-sha-256` + TLS
- [ ] Módulos de terceros revisados y dependencias auditadas
- [ ] Unidad systemd con puntuación de exposición < 3

Identidad y secretos

- [ ] MFA obligatorio para administradores y áreas sensibles
- [ ] Cuentas nominales, sin uso diario de `admin`
- [ ] Revisión trimestral de accesos documentada
- [ ] Sin secretos en claro ni en Git

Resiliencia y detección

- [ ] Copia inmutable con retención ⩾ 30 días
- [ ] Restauración completa probada este trimestre, dentro del RTO
- [ ] Logs enviados fuera del servidor
- [ ] auditd, AIDE y Fail2ban activos
- [ ] Alertas para administradores nuevos, cambios bancarios y fallos de copia
- [ ] Plan de respuesta con contactos y simulacro anual

15.3 Mantener el nivel

La seguridad se degrada sola: entran usuarios, se instalan módulos, se abren puertos "temporales". Programa una revisión mensual de la checklist, una auditoría externa o test de intrusión al año y vuelve al capítulo 1 cada vez que el ERP cambie de forma importante.

Ejercicios

1. Completa el proyecto en laboratorio y registra los indicadores: índice Lynis, exposición systemd, RTO medido.
2. Pasa la checklist a tu ERP de producción y planifica las casillas pendientes con fecha y responsable.

Apéndice: chuleta de comandos

Tarea	Comando
Ver puertos en escucha	`sudo ss -tulpn`
Auditoría general del sistema	`sudo lynis audit system`
Comprobar reglas nftables	`sudo nft -c -f /etc/nftables.conf`
Ver reglas activas	`sudo nft list ruleset`
Validar Nginx y recargar	`sudo nginx -t && sudo systemctl reload nginx`
Validar SSH	`sudo sshd -t`

Tarea	Comando
Exposición de un servicio	`systemd-analyze security odoo`
Logs de Odoo en vivo	`sudo journalctl -u odoo -f`
Roles de PostgreSQL	`sudo -u postgres psql -c '\du'`
Conexiones activas a la base	`sudo -u postgres psql -c 'SELECT usename, client_addr, state FROM pg_stat_activity;'`
Copia lógica de la base	`pg_dump -Fc -d erp_prod -f erp.dump`
Restaurar copia lógica	`pg_restore -d erp_nueva --no-owner erp.dump`
Copia de archivos cifrada	`restic backup /var/lib/odoo /etc/odoo`
Verificar repositorio de copias	`restic check`
Eventos de auditoría	`sudo ausearch -k erp_config -i`
Integridad de archivos	`sudo aide --check`
Estado de Fail2ban	`sudo fail2ban-client status odoo-login`
Desbloquear una IP	`sudo fail2ban-client set odoo-login unbanip 203.0.113.7`
Últimos inicios de sesión	`last -Fwx | head -20`
Intentos SSH fallidos	`sudo journalctl -u ssh --since today | grep Failed`
Buscar binarios SUID	`sudo find / -xdev -perm -4000 -type f`
Vulnerabilidades en dependencias Python	`pip-audit -r requirements.txt`
Secretos en un repositorio	`gitleaks detect`
Calidad TLS	`testssl.sh erp.ejemplo.com`

Contraportada

Tu ERP guarda la contabilidad, los clientes, las nóminas y las cuentas bancarias de la empresa. ¿Está protegido como lo que es?

Protección avanzada de sistemas ERP sobre Linux es una guía práctica para llevar un ERP de una instalación "que funciona" a una plataforma capaz de resistir intrusiones, ransomware, fraude interno y errores humanos.

Capa a capa, con configuraciones listas para adaptar, aprenderás a:

- Modelar las amenazas reales de un ERP y diseñar una arquitectura por zonas.
- Bastionar Linux, cerrar la red con nftables y publicar el ERP tras un proxy con TLS.
- Endurecer la aplicación ERP y su base de datos con mínimo privilegio, MFA y auditoría.
- Encerrar la aplicación con systemd, AppArmor/SELinux y contenedores seguros.
- Construir copias inmutables que sobrevivan a un administrador comprometido.
- Detectar, responder y cumplir con RGPD, ENS, NIS2 e ISO 27001.

Incluye ejercicios en cada capítulo, un proyecto final completo y una checklist de bastionado para auditar cualquier instalación.

Marlon Falcón Hernández es especialista en implantación y desarrollo de Odoo ERP y creador de la vertical de gestión de obra BIM 2.0. Desde Valencia, acompaña a empresas de España y Latinoamérica en la puesta en marcha y la protección de sus sistemas de gestión.

Implemento BIM 2.0 · Primera edición, 2026

Tu ERP guarda la contabilidad, los clientes, las nóminas y las cuentas bancarias de la empresa. ¿Está protegido como lo que es?

Protección avanzada de sistemas ERP sobre Linux es una guía práctica para llevar un ERP de una instalación «que funciona» a una plataforma capaz de resistir intrusiones, ransomware, fraude interno y errores humanos. Válida para Odoo, ERPNext, Dolibarr, Tryton, SAP o cualquier ERP sobre Linux.

Capa a capa, con configuraciones listas para adaptar, aprenderás a:

- Modelar las amenazas reales de un ERP y diseñar una arquitectura por zonas.
- Bastionar Linux, cerrar la red con nftables y publicar el ERP tras un proxy con TLS.
- Endurecer la aplicación ERP y PostgreSQL con mínimo privilegio, MFA y auditoría.
- Encerrar la aplicación con systemd, AppArmor/SELinux y contenedores seguros.
- Construir copias inmutables que sobrevivan a un administrador comprometido.
- Detectar, responder y cumplir con RGPD, ENS, NIS2 e ISO 27001.

Incluye ejercicios en cada capítulo, un proyecto final completo y una checklist de bastionado para auditar cualquier instalación.

Marlon Falcón Hernández

Especialista en implantación y desarrollo de Odoo ERP y creador de la vertical de gestión de obra BIM 2.0. Desde Valencia, acompaña a empresas de España y Latinoamérica en la puesta en marcha y la protección de sus sistemas de gestión.

Implemento BIM 2.0

Seguridad · Linux · ERP