

Biblioteca CROX

Vol. 2 — Edición Profesional

3 GUÍAS DE NIVEL AGENCIA

Hardening WordPress 2026 · Email Security Stack · Autauditoría en 60 minutos

Pago voluntario — XNO: nano_38ddjn4jd1pkgp6ugaoyyeqtkbaes7oby1lm1u8asfr7bwg13h5x51uky9r1
BTC: bc1qfulwc94ej036rauexs3aa76td0xpquhzld96lx
croxsec.surge.sh

Checklist Hardening WordPress 2026 — 40 controles priorizados

Guía profesional de endurecimiento WordPress para agencias y administradores.

Cada control incluye el riesgo que mitiga y el comando o paso exacto de implementación.

Orden sugerido: primero P0 (exposición activa), después P1 (superficie), final P2 (defensa en profundidad).

P0 — Exposición activa (hacer HOY)

1. Ocultar la versión de WordPress

El meta generator facilita ataques dirigidos a CVEs conocidos.

Quitar el filtro en `functions.php` del tema hijo:

```
remove_action('wp_head', 'wp_generator');  
add_filter('the_generator', '__return_empty_string');
```

Verificación: `curl -s https://tudominio.com | grep -i generator` → debe volver vacío.

2. Deshabilitar la edición de temas y plugins desde el admin

Un atacante con credenciales de editor no debe poder escribir PHP.

En `wp-config.php`:

```
define('DISALLOW_FILE_EDIT', true);
```

3. Forzar HTTPS con HSTS

En el virtual host nginx o `.htaccess` Apache:

```
add_header Strict-Transport-Security "max-age=31536000; includeSubDomains" always;
```

4. Bloquear el listado de directorios

En Apache: `Options -Indexes` en `.htaccess` raíz.

En nginx: `autoindex off;` (viene desactivado por defecto; verificar).

5. Proteger wp-config.php

Apache: regla en `.htaccess` raíz:

```
<files wp-config.php>
  order allow,deny
  deny from all
</files>
```

Nginx: `location = /wp-config.php { deny all; }`

6. Mover wp-config.php fuera del webroot (si la estructura lo permite)

WordPress busca `wp-config.php` un nivel arriba. Copiarlo a `/home/usuario/` (fuera de `public_html`) reduce el impacto de un LFI o backup expuesto.

7. Desactivar XML-RPC si no se usa

El 99 % de las instalaciones no lo necesita. Es vector de brute-force amplificado y pingback DDoS.

Apache `.htaccess`:

```
<Files xmlrpc.php>
  order allow,deny
  deny from all
</Files>
```

Verificación: `curl -d 'system.listMethods' https://tudominio.com/xmlrpc.php` → 403.

8. Cambiar el prefijo de tablas (en instalación nueva)

El prefijo `wp_` por defecto facilita SQLi automatizada. En `wp-config.php`:

```
$table_prefix = 'x7q_';
```

9. Proteger el login con rate limiting

Con fail2ban + regla nginx `limit_req`, o plugin ligero (Limit Login Attempts Reloaded). Bloqueo: 5 intentos / 15 min / IP.

10. Credenciales únicas y gestor

Nunca `admin` como usuario. Contraseñas de 20+ caracteres generadas. 2FA obligatorio para administradores (miniOrange o similar sin telemetría externa).

P1 — Superficie (esta semana)

11-14. Cabeceras de seguridad HTTP

Conjunto mínimo:

```
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;
```

CSP recomendada (ajustar a los assets reales):

```
add_header Content-Security-Policy "default-src 'self'; script-src 'self' 'unsafe-
```

15. Subir la caducidad de sesiones

En `wp-config.php`:

```
define('AUTH_KEY',          '...64 chars random...');
define('SECURE_AUTH_KEY',   '...');
define('LOGGED_IN_KEY',     '...');
define('NONCE_KEY',         '...');
define('AUTH_SALT',         '...');
define('SECURE_AUTH_SALT',  '...');
define('LOGGED_IN_SALT',    '...');
define('NONCE_SALT',        '...');
```

Generar en <https://api.wordpress.org/secret-key/1.1/salt/>

16. Restringir uploads a tipos seguros

Por defecto WordPress ya limita, pero verificar que no se permite `.php` en uploads: intentar subir `test.php` → debe fallar.

17. Desactivar el parseo de PHP en uploads

Crear `.htaccess` en `wp-content/uploads/`:

```
<Files *.php>
  deny from all
</Files>
```

18. Proteger wp-json de enumeración

`/wp-json/wp/v2/users` lista usuarios (facilita brute-force de nombres).

Plugin o filtro:

```
add_filter('rest_endpoints', function($endpoints) {
    if (isset($endpoints['/wp/v2/users'])) unset($endpoints['/wp/v2/users']);
    return $endpoints;
});
```

19. Borrar temas y plugins sin uso

Cada tema es código ejecutable potencial. Mantener: el activo + un tema fallback. Desactivar y ELIMINAR el resto (no basta con desactivar).

20. Actualizaciones automáticas menores

En wp-config.php:

```
define('WP_AUTO_UPDATE_CORE', 'minor');  
add_filter('auto_update_plugin', '__return_true');  
add_filter('auto_update_theme', '__return_false'); // revisar temas a mano
```

21. File permissions correctos

Verificación:

```
find /var/www -type d -exec chmod 755 {} \;  
find /var/www -type f -exec chmod 644 {} \;  
chmod 600 wp-config.php
```

22. Usuario de base de datos con privilegios mínimos

Crear usuario exclusivo para WP con solo los privilegios necesarios (SELECT, INSERT, UPDATE, DELETE, CREATE, ALTER, INDEX, DROP si es multisitio). Nunca root.

23. Base de datos con prefijo y nombre no obvios

miempresa_db y wp_ son los primeros intentos de herramientas automáticas.

24. Backups automáticos externos

Backup fuera del mismo servidor (S3 compatible, otro VPS, download semanal). Un backup en el mismo servidor es inútil ante ransomware. Verificar restauración trimestralmente.

25-28. WAF ligero (reglas nginx)

Bloquear user-agents vacíos, tráfico a wp-login.php desde IPs extrañas (geo-restricción si el cliente es local), scanner patterns en logs.

```
location /wp-login.php {  
    limit_req zone=login burst=2 nodelay;  
    include /etc/nginx/blocklists.conf; # IPs malas  
}
```

29. Desactivar el instalador de temas/plugins en producción

`DISALLOW_FILE_MODS` en wp-config impide instalar desde el admin (solo despliegue por git/rsync):

```
define('DISALLOW_FILE_MODS', true);
```

30. Registrar actividad (audit log)

Plugin Simple History o similar: saber QUIÉN cambió QUÉ y CUÁNDO. Retención: 90 días.

P2 — Defensa en profundidad (este mes)

31-34. Aislamiento

Si el servidor aloja varios sitios: cada sitio con usuario PHP-FPM propio (pool por sitio), `open_basedir` restringido, y sin acceso cruzado entre document roots.

35. Monitorización externa

Verificación diaria desde fuera: uptime, cambios de contenido en páginas clave (defacement), certificado TLS, aparición del dominio en listas de phishing. (Esto es exactamente lo que hace el Monitor CROX.)

36. Escaneo programado de malware

Maldet / ClamAV semanal + integración con fail2ban.

37. Política de passwords del equipo

Gestor compartido (Vaultwarden self-hosted), rotación al desvincular empleados, 2FA en todo.

38. Cabecera CSP progresiva

Empezar en modo report-only, afinar según reportes reales, luego enforcement.

39. Prueba de restauración trimestral

Documentar RTO/RPO reales: ¿cuánto tarda restaurar el último backup en un servidor limpio? Si no se ha medido, no existe.

40. Proceso de respuesta a incidentes

Una página: quién decide, quién ejecuta, quién comunica. Plantilla de comunicación a clientes. Lista de contactos (registrar, hosting, WAF).

Verificación externa final

Después de aplicar todo, escanear el sitio DESDE FUERA como lo haría un atacante:

cabeceras, exposiciones (/ .env /.git /wp-config.php.bak), enumeración de usuarios, TLS, cert.

Una mirada externa ve lo que la interna no ve.

Email Security Stack: SPF, DKIM y DMARC en producción

La suplantación de correo sigue siendo el vector #1 de compromiso de empresas pequeñas.

La solución no es un producto: son tres registros DNS correctamente configurados.

Esta guía lleva un dominio de cero a rechazo activo de suplantación, con verificación en cada paso.

El problema en una frase

Cualquiera puede enviar un correo "desde" tu dominio porque SMTP no valida remitentes.

SPF, DKIM y DMARC son las tres piezas que cierran ese agujero, en ese orden de despliegue.

Fase 0 — Inventario (antes de tocar nada)

Antes de publicar registros, saber QUÉ servicios envían correo como tu dominio:

- Servidor de correo propio (MX)
- Webmail del hosting
- Proveedor de newsletters (Mailchimp, Brevo, Mailgun, SES...)
- CRM / helpdesk
- Scripts del servidor (formularios web, avisos, backups)
- Google Workspace / Microsoft 365

Comando para ver el estado actual:

```
dig +short MX tudominio.com
dig +short TXT tudominio.com | grep spf
dig +short TXT _dmarc.tudominio.com
dig +short TXT google._domainkey.tudominio.com # si usas Google
```

Pasar cada servicio por una checklist: ¿envía? ¿desde qué IPs? ¿qué selector DKIM usa?

Fase 1 — SPF (quién puede enviar)

SPF publica la lista de IPs/servidores autorizados.

Registro mínimo para un dominio que solo envía desde su hosting:

```
v=spf1 mx -all
```

Con servicios externos (ejemplo con Google Workspace + Mailchimp):

```
v=spf1 mx include:_spf.google.com include:servers.mcsv.net -all
```

Reglas duras:

- SOLO UN registro SPF por dominio. Dos registros = ambos se ignoran en muchos receptores.
- Terminar en `-all` (rechazo duro) cuando el inventario esté completo. `~all` (softfail) solo durante el periodo de prueba.
- El límite de 10 búsquedas DNS se alcanza rápido con includes anidados. Usar `ip4:/ip6:` directos cuando sea posible.
- No usar `+all`. Nunca.

Verificación:

```
dig +short TXT tudominio.com | grep spf  
<h1>enviar test a mail-tester.com y revisar la nota SPF</h1>
```

Fase 2 — DKIM (el correo lleva firma criptográfica)

DKIM firma cada correo saliente con la clave privada del servidor; el receptor valida contra la pública en DNS.

Pasos generales (cada proveedor tiene su panel):

1. Generar par de claves (2048 bits mínimo hoy).
2. Publicar la pública como TXT: `selector._domainkey.tudominio.com`.
3. Activar la firma en el servidor/proveedor.

Ejemplo de registro (Google usa `google`, otros usan `selector1`, `s1`, `mail`...):

```
google._domainkey IN TXT "v=DKIM1; k=rsa; p=MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIJ
```

Verificación de una firma real:

```
<h1>enviarse un correo a una dirección propia y mirar las cabeceras:</h1>  
<h1>Authentication-Results: ... dkim=pass header.i=@tudominio.com header.s=google-
```

Error común: rotar la clave y no actualizar DNS → correos firmados con la privada vieja que ya no validan. Planificar rotación anual con solape.

Fase 3 — DMARC (la política de qué hacer cuando algo falla)

DMARC dice al receptor: "si SPF o DKIM fallan con mi dominio, rechaza/cuarentena, y mándame el reporte".

Despliegue gradual — el orden importa:

Semana 1-2: solo observar

```
_dmarc IN TXT "v=DMARC1; p=none; rua=mailto:dmarc@tudominio.com; fo=1; adkim=s;
```

`p=none` no rechaza nada: solo llegan reportes XML diarios de quién envía como tu dominio.

Semana 3-4: cuarentena al 50%

```
"v=DMARC1; p=quarantine; pct=50; rua=mailto:dmARC@tudominio.com; fo=1; adkim=s; as
```

Después: rechazo total

```
"v=DMARC1; p=reject; rua=mailto:dmARC@tudominio.com; ruf=mailto:dmARC-forense@tud
```

Parámetros clave:

- `adkim=s` / `aspf=s`: alineación estricta (el dominio del From: debe coincidir exactamente). Es lo que realmente para el phishing de lookalikes.
- `fo=1`: reporte forense si falla cualquier mecanismo.
- `rua`: reportes agregados diarios (mailbox que alguien LEERÁ).
- `p=reject`: el objetivo final. Sin él, un atacante seguirá llegando a bandejas.

Fase 4 — MTA-STS y TLS-RPT (bonus profesional)

MTA-STS obliga a los remitentes a usar TLS al entregarte correo:

```
_mta-sts IN TXT "v=STSV1; id=20261028T000000;"
```

+ un archivo `https://mta-sts.tudominio.com/.well-known/mta-sts.txt`:

```
version: STSV1
mode: enforce
mx: tu-servidor-mail.com
max_age: 604800
```

TLS-RPT (`_smtp._tls` TXT) recibe reportes de conexiones que cayeron por TLS.

Verificación completa (checklist final)

<h1>1. SPF publicado y único</h1>

dig +short TXT tudominio.com | grep -c "v=spf1" # debe dar 1

<h1>2. DKIM valida en correo real (mirar cabeceras Authentication-Results)</h1>

<h1>3. DMARC con política de rechazo</h1>

dig +short TXT _dmarc.tudominio.com

<h1>4. Test de suplantación controlado</h1>

<h1>Enviar desde un servidor externo con From: @tudominio.com hacia Gmail</h1>

<h1>Resultado esperado: 550 5.7.1 (rejected by DMARC)</h1>

<h1>5. mail-tester.com → nota 10/10</h1>

Errores que destruyen la entregabilidad

- Dos registros SPF (se anulan mutuamente).
- `p=reject` sin haber pasado por `p=none` con reportes (el propio correo legítimo empieza a rebotar).
- Includes SPF encadenados que superan las 10 búsquedas.
- Selector DKIM incorrecto tras migrar de proveedor.
- No leer los reportes DMARC: ahí se ve el correo legítimo olvidado (un script viejo, una newsletter) antes de que el rechazo lo rompa.

El estándar CROX

Un dominio corporativo está bien configurado cuando:

SPF termina en `-all`, DKIM firma con 2048 bits, DMARC en `p=reject` con alineación estricta,

y existe alguien que lee los reportes semanales.

Coste: 0 €. Tiempo: 2-4 semanas de despliegue gradual. Efectividad contra suplantación directa: casi total.

Autauditoría externa de tu empresa en 60 minutos

La misma metodología que usamos en CROX Sec, condensada para que la ejecutes tú mismo hoy.

Sin herramientas de pago, sin instalar nada, sin tocar tus sistemas: todo es recon pasivo.

Al final tendrás una lista priorizada de lo que un atacante vería primero.

Reglas del ejercicio

- 100 % pasivo: consultas DNS y HTTP de lectura. Cero escaneo agresivo.
- Desde una red que no sea la de la oficina (datos móviles) — simula al atacante real.
- Anota cada hallazgo con: qué viste, dónde, y qué haría un atacante con ello.

Minuto 0-10 — Inventario de superficie

Dominios y subdominios

```
<h1>Registros del dominio raíz</h1>
dig +short MX midominio.com
dig +short NS midominio.com
dig +short TXT midominio.com

<h1>Subdominios vía Certificate Transparency (lo que la gente olvida listar)</h1>
curl -s "https://crt.sh/?q=%25.midominio.com&output=json" | \
  python3 -c "import sys,json; d=json.load(sys.stdin); print('\n'.join(sorted({n['
```

Lo que aparece aquí y NADIE reconoce = subdominio olvidado = superficie sin parchear.

Comprobar cada uno: ¿resuelve? ¿qué servicio corre? ¿tiene TLS válido?

IPs y hosting

```
dig +short A midominio.com  
whois <IP> | grep -iE "org-name|netname|country"
```

Saber dónde se aloja define parte de la superficie (cloud compartido, VPS dedicado, on-prem).

Minuto 10-25 — TLS y cabeceras

TLS

```
curl -sI https://midominio.com | head -5  
<h1>o mejor:</h1>  
openssl s_client -connect midominio.com:443 -servername midominio.com < /dev/null
```

Checklist:

- ¿Caduca en menos de 30 días? → proceso de renovación frágil.
- ¿Emisor desconocido/mismo servidor? → cert autofirmado en algún servicio interno.
- ¿TLS 1.0/1.1 aceptado? → obsoleto y vulnerable.

Cabeceras de seguridad

```
curl -sI https://midominio.com
```

Ausentes típicos: Strict-Transport-Security, Content-Security-Policy, X-Content-Type-Options,

X-Frame-Options, Referrer-Policy, Permissions-Policy.

Cada ausencia es una categoría de ataque que no está mitigada (clickjacking, MIME sniffing, XSS inyectado...).

Minuto 25-40 — Exposiciones clásicas

Probar las rutas que TODO atacante prueba primero:

```
for p in /.env /.git/HEAD /.git/config /config.php /wp-config.php.bak /backup.zip
do
  code=$(curl -s -o /dev/null -w "%{http_code}" "https://midominio.com$p")
  echo "$p -> $code"
done
```

Interpretación:

- 200 en /.env o /.git → CRÍTICO: credenciales o código fuente expuestos. Arreglar hoy.
- 200 en backup.* → CRÍTICO: descarga y contiene probablemente la base de datos.
- 301/302 en /phpinfo.php → buscar dónde aterriza.
- 404 en security.txt → escribirlo: es gratis y los investigadores serios lo usan.

Rutas específicas por stack:

- WordPress: /wp-json/wp/v2/users, /xmlrpc.php, /wp-content/debug.log
- phpMyAdmin: /phpmyadmin, /pma, /mysql
- Paneles: /admin, /cpanel, :8080, :2082, :10000 (Webmin)

Minuto 40-50 — Correo: la puerta grande

```
dig +short TXT midominio.com | grep spf
dig +short TXT _dmarc.midominio.com
dig +short TXT google._domainkey.midominio.com # o el selector del proveedor
```

- Sin SPF -> cualquiera puede enviar como tu dominio. Prioridad ALTA.
- SPF con +all o ~all → débil.
- Sin DMARC → nadie rechaza la suplantación. Prioridad ALTA.

- DMARC p=none → solo mira; no protege.

Test de suplantación autorizado (sólo contra tu propio dominio, un correo de prueba):

envíate un correo desde un servidor externo con el From: de tu dominio hacia una bandeja Gmail

de prueba. Si llega a bandeja: SPF+DMARC fallan.

Minuto 50-60 — Filtraciones públicas y huella

```
<h1>¿Emails corporativos en breaches conocidos? (HaveIBeenPwned vía API web)</h1>
<h1>¿Credenciales de empleados en colecciones públicas? búsqueda de "@midominio.co
<h1>- IntelX, leakcheck, dehashed (algunos gratuitos con límite)</h1>
<h1>Contraseñas filtradas de UN empleado = entrada inicial.</h1>

<h1>Fingerprinting del stack</h1>
curl -sI https://midominio.com | grep -iE "server|x-powered"
whatweb https://midominio.com # si está instalado
```

Versiones antiguas visibles = CVEs conocidos listos para explotar.

Salida: tu lista priorizada

Clasifica cada hallazgo:

| Prioridad | Criterio | Ejemplos | |---|---|---| | P0 | Exposición de credenciales/código | .env, .git, backups, phpinfo | | P1 | Suplantación posible | sin SPF/DMARC, DMARC p=none | | P2 | Superficie sin mitigar | cabeceras ausentes, TLS viejo, subdominios olvidados | | P3 | Higiene | security.txt, versiones visibles, paneles expuestos |

Qué hacer con la lista

- P0 hoy, antes de dormir.

- P1 esta semana (guía Email Security Stack de esta biblioteca).

- P2-P3 en el sprint de hardenizado mensual.

Cuándo externalizar esto

La autoauditoría cubre el 70 % de la superficie obvia. El 30 % restante requiere:

monitoreo continuo (los cambios llegan solos: un subdominio nuevo, un certificado que caduca,

un .env que alguien sube un viernes), correlación de CT logs en el tiempo, y ojos que ven lo

que una checklist no ve. Ahí es donde un servicio como el Monitor de Postura aporta:

mismo método, pero ejecutado cada día y con historial.

Documento CROX Sec — reproducible, verificable, sin humo.

Si encuentras un error técnico en esta guía: crox@croxsec.com (lo corregimos y te mencionamos).