WPdanger

Guía de Seguridad para WordPress

Javier Casares

WPdanger:
Guía de Seguridad para WordPress

Índice

Licencia

[1] https://eupl.eu/1.2/es/

Comenzando con la seguridad en WordPress

WordPress es un sistema seguro, de código abierto y mantenido por la comunidad lo que lo hace fácilmente actualizable en cuanto se detecta cualquier problema.

Y es que WordPress es un sistema compuesto por muchos elementos, algunos desarrollados por terceros, que hay que ir actualizando y manteniendo.

En este documento se van a dar soluciones sencillas (aunque a veces complejas de implementar) que han de servir de base para tu configuración y mantenimiento. Los plugins, plantillas o soluciones que se dan te han de servir de guía por lo que, en la mayoría de casos, te recomendamos contactar con un Administrador de Sistemas (*SysAdmin*) que ayude con las configuraciones más complejas.

IMPORTANTE: Si no sabes lo que haces, es mejor que no toques.

Ataques que puedes sufrir

Existen infinidad de posibles ataques a tu sitio, ya que existen infinidad de posibles puntos de acceso a los mismos, ya sean por el propio WordPress, pero también por el servidor, los usuarios, comentarios, códigos de script, etcétera. Algunos de los más habituales pueden ser estos:

- Authentication Bypass: Agujero de seguridad que permite saltarse el formulario de acceso y acceder al sitio.
- Brute Force: Se intenta iniciar sesión adivinando el nombre de usuario y la contraseña de la cuenta de administrador (o de un usuario).
- Cross-Site Request Forgery (CSRF): El código se introduce y ejecuta desde la URL.
- Cross-site Scripting (XSS): Se puede inyectar código en un sitio, normalmente a través de un campo de formulario.
- Denial of Service (DoS): Un sitio se cae debido a un ataque constante de tráfico que suele proceder de una red de máquinas controladas.
- Path Traversal: Posibilidad de listar los directorios de un sitio y ejecutar comandos fuera del directorio raíz del servidor.
- Distributed Denial of Service (DDoS): Similar a un ataque DoS, excepto que las redes de máquinas suelen haber sido infectadas.
- File Upload: Se puede subir un fichero con código malicioso en un servidor sin restricciones.
- Full Path Disclosure (FPD): Se expone la ruta de acceso a la carpeta raíz del sitio; habitualmente es debido a que están activos los mensajes de error que las muestran.
- Local File Inclusion (LFI): Un atacante es capaz de controlar qué archivo se ejecuta en una hora programada que fue configurada anteriormente.
- Malware: Un sitio o programa malintencionado con el propósito de infectar al usuario u otra máquina.
- Open Redirect: El sitio redirige a otro debido a alguna vulnerabilidad, a menudo un sitio de spam o de suplantación de identidad.
- Phishing (Identity Theft): Un sitio que se parece a otro conocido y de confianza, pero que se utiliza para recopilar credenciales de inicio de sesión, números de tarjetas de crédito, etcétera, engañando al usuario.
- Remote Code Execution (RCE): Capacidad de ejecutar código en un sitio desde una máquina diferente.

- Remote File Inclusion (RFI): Posibilidad de ejecutar un script externo en un sitio al que se suele cargar malware, desde un sitio diferente.
- Security Bypass: Similar al Authentication Bypass, pero en este caso permite saltarse algún sistema de seguridad establecido.
- Server-side Request Forgery (SSRF): Toma de control de un servidor, ya sea parcial o total, para obligarlo a ejecutar peticiones de forma remota.
- SQL Injection (SQLI): Se produce cuando consultas SQL se pueden introducir y ejecutar desde la URL de un sitio.
- User Enumeration: Posibilidad de encontrar la lista de usuarios de un sitio desde la zona pública, para posteriormente realizar un ataque de Brute Force.
- XML External Entity (XXE): Un fichero XML que por la generación de errores deja expuesto algún tipo de ruta, mensaje o acceso a información confidencial.

Tipos de alojamiento para tu WordPress

Existen muchas formas de tener tu WordPress disponible en línea. Desde alojamientos web compartidos, dedicados, virtuales, en la nube… Cada uno de ellos tiene sus ventajas e inconvenientes. Ten presente que en este documento se presentan muchas opciones que sólo estarán disponibles si tienes control de toda la máquina y de su configuración. En caso contrario deberías poder contactar con tu proveedor y solicitar dichos cambios por el bien de tu sitio. En caso de que no los quieran aplicar, deberías plantearte si esa es la opción que necesitas.

En cualquier caso, tus necesidades de alojamiento web serán cada vez más complejas según la complejidad y crecimiento de tu sitio, y por extensión, de su seguridad.

Tu alojamiento web es como la tienda física de tu negocio. No es lo mismo tener tu tienda física en el centro de una ciudad o en un centro comercial que en las afueras. Cada uno de estos lugares tienen sus elementos a favor y sus inconvenientes. Elige correctamente tu alojamiento web ya que influirá en gran medida en la seguridad de tu sitio.

Mantén tu reloj en hora

Estar al día con la hora de tu servidor y de tu sitio web es importante por muchas razones. Por un lado, porque si hay que revisar “logs”, saber a qué hora exacta ocurren las cosas es importante. De la misma forma si usas herramientas de verificación es posible que sincronicen los datos de tu sitio web con los de tu móvil (por ejemplo). Recuerda que los servidores es mejor que siempre estén configurados en hora UTC, y luego podrás configurar tu WordPress con el huso horario que quieras.

Mantén limpia tu casa

Por supuesto que vamos a hablar de WordPress y mucho, pero también es importante saber sobre qué se va a montar WordPress, qué servidor, qué versiones, qué soporte y hasta cuándo estará…

La primera pregunta es: ¿cuál es el mejor sistema operativo para WordPress? La respuesta es simple y compleja: Linux. Aquí entraríamos en cuáles son las mejores distribuciones de Linux y entraríamos en una batalla demasiado antigua, por lo que me voy a limitar a decir que una distribución que mantenga actualizado su repositorio. Como experiencia personal, yo he trabajado muchos años con CentOS (RedHat) y

últimamente me he pasado a Ubuntu. En este último caso, tomé la decisión de siempre usar las últimas versiones LTE estables (hoy en día es la rama versión 16).

La segunda pregunta es: ¿cuál es el mejor servidor web para WordPress? De nuevo la respuesta es simple y compleja: o Apache o nginx; preferiblemente el segundo. Una vez más está basado en mi experiencia de haber usado Apache durante más de una década y haber descubierto la sencillez de nginx en determinados casos. Con WordPress prefiero usar nginx en su última versión estable.

La tercera pregunta es: ¿cuál es la mejor base de datos para WordPress? Y una vez más, la respuesta es sencilla: MySQL, y como tal, comienza la aventura de qué *fork* es el ideal. En principio yo he probado el MySQL de Oracle, el Percona MYSQL y MariaDB. Hasta ahora en general los tres se parecían bastante en cuanto a funcionalidad, en principio Percona y MariaDB daban mejor rendimiento y últimamente comienzan a haber diferencias entre ellos. Por mi experiencia, últimamente yo utilizo MariaDB (al menos su versión 10.0).

Y la última pregunta: ¿PHP? Sí, como lenguaje de programación no hay ninguna otra opción. Hace ya algunas versiones se impuso como mínimo el uso de la 5.5, y en las últimas ya se comienza a presionar para ir avanzando en el uso de versiones más modernas. Por un tema del funcionamiento de PHP, ahora mismo hay tres opciones claras[2]: 5.6, 7.0 y 7.1. En este caso sí que usaré la página de la propia WordPress de requerimientos[3] y diré claramente que 7 o superior.

Con respecto a este punto, existe una herramienta que te puede ayudar a tomar la decisión de si actualizar a una versión de PHP superior. Se llama PHP Compatibility Checker[4] y básicamente cuando lo activas analiza tus plantillas y mejoras y te dará información sobre su compatibilidad o no.

En general recomiendo dar una ojeada a esta página de la propia WordPress en las que se va actualizando los mínimos de funcionamiento, aunque lo ideal es usar las últimas versiones estables de todo.

Un detalle interesante es observar qué versiones se utilizan en la mayoría de instalaciones de WordPress por el mundo[5] aunque, personalmente, no es para nada ninguna guía, porque veo demasiadas instalaciones desactualizadas.

Mantén WordPress actualizado

WordPress lleva internamente su propio sistema de actualización que incluye el núcleo (*core*), los elementos añadidos (*plugins*), las plantillas de diseño (*themes*) y las traducciones de todos ellos (*translations*).

Al menos una vez a la semana deberías entrar en la sección [Actualizaciones] y aplicar las mejoras que te sugiere el sistema, que suelen haber sido probadas por un centenar de usuarios avanzados anteriormente.

Además, si quieres que el sistema se actualice de forma automática puedes hacerlo añadiendo unas líneas de código al fichero de configuración [wp-config.php]:

[2] http://php.net/supported-versions.php
[3] https://wordpress.org/about/requirements/
[4] https://wordpress.org/plugins/php-compatibility-checker/
[5] https://wordpress.org/about/stats/

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

En algunos casos, por las distintas configuraciones y posibles incompatibilidades, es posible que no se deban realizar las actualizaciones de forma automática, sino únicamente de forma manual. En este caso deberíamos añadir al fichero de configuración [wp-config.php] la siguiente línea:

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

Otra posibilidad es la de forzar la actualización de todo mediante la creación de un elemento añadido (*plugin*) obligatorio. Para ello subiremos -por FTP, SSH- el plugin [WPdanger Autoupdater][6] a la carpeta [/wp-content/mu-plugins/]. Este plugin se activará siempre en el sitio y obligará a actualizar todo de forma automática.

¿Cuándo se actualizó ese complemento o plantilla?

Está muy bien actualizar los complementos (*plugins*) o las plantillas (*themes*) pero ¿por qué instalas un complemento que hace más de un año que no se actualiza?

Está claro que hay plugins que seguramente por su funcionalidad no requieren de grandes cambios, pero simplemente por mantener las fechas y los datos de testeo actualizados, cualquier desarrollador de un *plugin* o *theme* debería actualizar los datos.

Y es que todos los complementos llevan interiormente un dato que debería estar al día: versión "hasta la que se ha probado". Si van apareciendo versiones y versiones de WordPress y el programador del plugin no va probando en esas nuevas versiones su funcionamiento ¿quién lo va a hacer?

Esta nueva versión no me funciona bien

Volver atrás con una versión del núcleo de WordPress es una mala idea. Si vas manteniendo WordPress de forma habitual, seguramente si tienes problemas no serán por culpa del sistema, sino de alguna plantilla o complemento. Es ahí donde sí que puedes tener problemas para volver atrás. En estos casos en los que hayas actualizado a una versión superior y tengas problemas, puedes probar de hacer un *rollback* (volver atrás) con WP Rollback[7]. Este sistema te permitirá elegir qué versión elegir de una plantilla o complemento, y así gestionarte tú mismo de una manera más controlada. Aún así, desde aquí te recomendamos que en caso de tener cualquier problema de incompatibilidad contactes con el autor para que intente corregir el problema que puedas tener, ya que si tú lo tienes, es casi seguro que muchos otros también.

Configuración de permisos de ficheros

Cuando se realiza una instalación nueva de WordPress, un elemento a tener presente es quién es el propietario de los ficheros del sistema y qué permisos tienen para poderse leer o escribir. Esto es importante porque en momentos de un posible *hackeo* se podrán extender los problemas más o menos.

[6] http://bit.ly/wpdanger-autoupdater
[7] https://wordpress.org/plugins/wp-rollback/

Lo primero es asegurarse qué usuario va a ser el que acceda a los ficheros y a qué grupo pertenece. Por norma general los sistemas vienen configurados para Apache HTTPD[8] o nginx[9], por lo que podemos configurarlo para ellos. Se puede ejecutar este comando por SSH desde la carpeta de instalación del WordPress.

En Apache HTTPD:

```
chown -R apache:apache ./
```

En nginx:

```
chown -R www-data:www-data ./
```

Posteriormente hay que dar los permisos de [lectura / escritura / ejecución] correspondientes a los ficheros. En general daremos permisos 755 a las carpetas, y 644 a los ficheros.

```
find /var/www/html/ -type d -exec chmod 755 {} \;
find /var/www/html/ -type f -exec chmod 644 {} \;
```

Si el sistema está bien configurado, podría ser un poco más agresivo y bloquear aún más accesos externos de usuarios indeseados. En este caso dejaremos siempre la última de las tres cifras a cero.

```
find /var/www/html/ -type d -exec chmod 750 {} \;
find /var/www/html/ -type f -exec chmod 640 {} \;
```

Una vez hayamos dado estos permisos a todo el WordPress, haremos un par de excepciones con dos ficheros muy concretos: el [wp-config.php] y el [readme.html].

El primero de ellos hemos de dejar que el sistema sea capaz de leerlo, pero ya está, nadie más debería cambiar o acceder.

```
chmod 600 wp-config.php
```

Si queremos ser algo más agresivos y que ni el propio sistema sea capaz de modificarlo, podemos ejecutar el comando:

```
chmod 400 wp-config.php
```

En el primer caso el fichero podría leerse y escribirse; en el segundo sólo leerse. Esto podría hacer que, si tu WordPress necesita hacer algún cambio en el fichero, no se pueda llegar a aplicar y lo tengas que modificar manualmente.

El segundo es un fichero que ofrece una pequeña situación con la seguridad, la versión del WordPress. Es el fichero más sencillo para detectarla y no aporta nada más al sitio, por lo que bloquearemos su lectura, dejando sólo que se pueda escribir cuando haya actualizaciones.

[8] https://httpd.apache.org/
[9] https://nginx.org/

```
chmod 200 readme.html
```

Para finalizar, de la misma forma que hemos cambiado los permisos en los ficheros, vamos a bloquear su acceso de forma pública por los usuarios web. Así, el sistema será capaz de leerlos o actualizarlos, pero no serán navegables.

En Apache HTTPD (dentro del fichero .htaccess):

```
<files wp-config.php>
  deny from all
</files>
<files readme.html>
  deny from all
</files>
<files license.txt>
  deny from all
</files>
```

En nginx (dentro del fichero de configuración del sitio):

```
location ~ \.php$ {
  location ~ wp-config {
    deny all;
  }
  location ~ /xmlrpc.php {
    limit_except POST {
      deny all;
    }
  }
}
location ~* readme.(html|txt) {
  deny all;
}
location ~* ^license.txt {
  deny all;
}
```

Para acabar, un detalle que puede venir de serie activo en los servidores Apache HTTPD es el listado de los contenidos de los directorios. Para evitar esto, y que no se muestren los ficheros de algunas carpetas como la de [uploads], debemos desactivar una opción en el .htaccess.

En Apache HTTPD (dentro del fichero .htaccess):

```
Options -Indexes
```

Seguridad tras la primera instalación

Esta configuración que te presento se puede tener configurada previa a la instalación de un nuevo WordPress, pero hay un elemento que necesitarás en la instalación y que luego tendrás que bloquear. Es el fichero de creación de la Configuración [/wp-admin/setup-config.php] y el de Instalación [/wp-admin/install.php].

Cuando acabes de instalar tu WordPress por primera vez, ya no necesitarás este fichero, y nadie tampoco debería de poder acceder, así que limitaremos su acceso:

```
location ~* ^/wp-admin/(install|setup-config).php {
  deny all;
}
```

Deshabilitar la edición de ficheros

Las personas, en general, somos curiosas por naturaleza y eso hace que, si desde el panel de administración se puede tocar algo, lo toquemos. Esto, en la parte de las plantillas, puede implicar que "algo deje de funcionar". Para ello existe la posibilidad de bloquear la edición de ficheros y que, en caso de ser necesario, se tenga que hacer vía FTP o SSH. Para ello añadiremos la siguiente línea el fichero de configuración [wp-config.php]:

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

Además, podemos ser un poco más agresivos y bloquear la posibilidad de que se puedan añadir plantillas (*themes*) o extensiones (*plugins*) desde el panel de administración. Para ello añadiremos la siguiente línea el fichero de configuración [wp-config.php]:

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

Cambio de las carpetas por defecto

Cuando hablas de WordPress es un clásico hablar del "uvedoblepé content" [/wp-content/] haciendo referencia a la carpeta por defecto donde se encuentran plantillas, ficheros, etcétera. Esta es una forma sencilla de detectar si tu instalación tiene WordPress; pero el sistema permite cambiar estas carpetas por otros nombres que tú quieras. Para ello puedes modificar algunos elementos en el fichero de configuración [wp-config.php]:

- Carpeta de contenidos [/wp-content/]

```
define('WP_CONTENT_DIR', '/contenidos');
define('WP_CONTENT_URL', 'https://www.example.com/contenidos');
```

- Carpeta de *plugins* [/wp-content/plugins/]

```
define('WP_PLUGIN_DIR', '/contenidos/mejoras');
define('WP_PLUGIN_URL', 'https://www.example.com/contenidos/mejoras');
```

- Carpeta de plantillas (*themes*) [/wp-content/themes/]

```
$theme_root = WP_CONTENT_DIR.'/plantillas';
define('UPLOADS', 'contenidos/plantillas');
```

Accesos al panel de administración

Aunque podemos cambiar las carpetas por defecto del sistema, no es posible cambiar las de la administración [/wp-admin/]. Hay dos opciones en este caso.

La primera y más habitual es la que se suele sugerir e bloquear el acceso a esta carpeta limitada a una serie de IP, pero también sabemos que en general (al menos con IPv4) los usuarios no suelen tener una IP fija para acceder por ahí. Así que si tienes IP fija, podrías configurarlo, pero si no, esta opción no es tan viable.

En Apache HTTPD (dentro del fichero .htaccess en la carpeta /wp-admin/) puedes limitar tu conexión a una IP en concreto:

```
order deny, allow
allow from 8.8.8.8
deny from all
```

Otra posibilidad sería la de mostrar un mensaje de acceso de usuario de sistema operativo (que no dependa de WordPress) pudiendo bloquear, por ejemplo, las peticiones a la página de /wp-login.php:

```
<Files wp-login.php>
  AuthUserFile ~/.htpasswd
  AuthName "Acceso privado bajo llave"
  AuthType Basic
  require user miusuariosecreto
</Files>
```

De la misma forma, en nginx se puede configurar lo siguiente:

```
location /wp-admin {
  allow 8.8.8.8;
  deny all;
}
```

Y también se podría configurar bajo una solicitud de contraseña:

```
location /wp-login.php {
  auth_basic "Acceso privado bajo llave";
  auth_basic_user_file .htpasswd;
}
```

Si quieres tener varios usuarios y contraseñas, deberás configurar el fichero .htpasswd con los accesos según convenga.

La segunda es la de utilizar un plugin que te permita cambiar la dirección URL, más similar a los cambios anteriores. Uno de ellos es WPS Hide Login[10], que te permitirá cambiar el acceso.

Ten presente que en cualquiera de los dos casos se generan una serie de problemas asociados. Para comenzar las cachés; si utilizas algún sistema deberás configurarlo para que no cachee esta nueva dirección. Por otro están las llamadas AJAX asíncronas

[10] https://wordpress.org/plugins/wps-hide-login/

que se suelen hacer a ficheros de esta carpeta: si bloqueas la carpeta a una IP o cambias el nombre, es posible que pierdas toda esta funcionalidad.

Para evitar esto, en Apache HTTPD deberías desbloquear el acceso al fichero:

```
<Files admin-ajax.php>
  Order allow, deny
  Allow from all
  Satisfy any
</Files>
```

Y de la misma forma, en nginx:

```
location /wp-admin/admin-ajax.php {
  allow all;
}
```

Bloquear PHP en carpetas de media

En principio, cuando subes ficheros mediante el panel de Multimedia sólo se permiten elementos como imágenes, ficheros de texto y similares, pero no se permiten ficheros PHP (el lenguaje de programación con el que se hace WordPress) entre otros. Si esto estuviera activo, podría permitir que un *hacker* pueda ejecutar procesos maliciosos que no se deben ejecutar. Para ello deberíamos bloquear la posibilidad de ejecutar ficheros PHP en esta carpeta.

En Apache HTTPD (dentro del fichero `.htaccess` en la carpeta [`/wp-content/uploads/`]):

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

Si queremos ser más agresivos, podemos bloquear muchos más lugares donde no es necesario que el público tenga que acceder, y que el núcleo de WordPress sí:

En nginx (dentro del fichero de configuración del sitio):

```
location ~* /wp-includes/.*\.php$ {
  deny all;
  access_log off;
  log_not_found off;
}
location ~* /wp-content/.*\.php$ {
  deny all;
  access_log off;
  log_not_found off;
}
location ~* /(?:uploads|files)/.*\.php$ {
  deny all;
  access_log off;
```

```
  log_not_found off;
}
```

Acceso a la base de datos

Con respecto a la base de datos (MySQL, MariaDB…) hay pocas cosas a hacer propiamente dichas desde el punto de vista de WordPress, pero sí que se pueden poner algunas complicaciones para aquellos que intenten acceder a ella.

La primera es la del acceso a la misma. Por norma general WordPress se instala en máquinas que pueden tener acceso externo, por lo que hemos de bloquearlo.

Para ello, cuando creemos el usuario, hemos de limitarlo para que, por ejemplo, sólo se pueda acceder desde la misma máquina si tienes todo en un mismo servidor:

```
GRANT ALL ON tu_database.* TO 'tu_usuario'@'localhost' IDENTIFIED BY 'tu_contraseña';
```

Con esto, tu usuario sólo tendrá acceso a una base de datos y sólo desde la misma máquina. Así, si tienes WordPress y te roban los accesos de una web, no podrían acceder a los de otra máquina.

Además, si tienes tu servidor web en una máquina y tu base de datos en otra, puedes aplicar una configuración similar (ejemplo con IP privadas):

```
GRANT ALTER, CREATE, DELETE, DROP, INDEX, INSERT, SELECT, UPDATE ON tu_database.* TO
'tu_usuario'@'10.0.0.2' IDENTIFIED BY 'tu_contraseña';
```

O con IP públicas:

```
GRANT ALTER, CREATE, DELETE, DROP, INDEX, INSERT, SELECT, UPDATE ON tu_database.* TO
'tu_usuario'@'8.8.8.8' IDENTIFIED BY 'tu_contraseña';
```

De esta forma sólo las máquinas con esas IP podrían acceder a tu base de datos. Ten presente que todo esto se puede complicar mucho y que estos son ejemplos sencillos pero útiles. Úsalos como base para tu configuración personalizada.

Hay algunos detalles más que se pueden deshabilitar en determinados casos. Por ejemplo, si tu base de datos está en la misma máquina que el sitio web (o sea, el servidor es “localhost”) podrías desactivar (o al menos comprobar) que no funcionaría ningún acceso externo. Para ello puedes buscar el fichero de configuración de MySQL en [/etc/my.cnf] o [/etc/msqyl/my.ini], y verificar que en la zona de configuración [[mysqld]] tienes incluidas estas dos líneas:

```
skip-networking
bind-address=127.0.0.1
```

Además, algo que seguramente nunca harás con tu WordPress y su base de datos es cargar ficheros en la misma, por lo que desactivaremos los comandos de LOCAL INFILE, en la misma zona que el punto anterior:

```
set-variable=local-infile=0
```

Otro de los elementos clásicos de seguridad en WordPress es el "prefijo" de las tablas de la base de datos. Este sistema permite que cohabiten varios WordPress en una única base de datos. Por defecto el prefijo histórico de WordPress es el [wp_], que deberíamos cambiar.

Lo más habitual es poner un nombre sencillo. Si mi web es example.com, lo más probable es que ponga como prefijo [example_]; esto es mejor que el caso anterior, pero algo bastante fácil de encontrar. Es por esto que te recomiendo utilizar nombres más aleatorios del estilo a [a1b2c3_] en el que cruces letras y números. Recuerda sólo utilizar letras en minúsculas de la [a] a la [z], y números del [0] al [9]. Por poner algunos ejemplos más: [c2712m_], [jgriyz_], [ah8zhl_]…

Cambiaremos el nombre de los prefijos en el fichero de configuración [wp-config.php]:

```
$table_prefix = 'wp_';
```

¿Qué ocurre si mi instalación de WordPress es antigua y tengo ya muchas tablas y configuraciones? Puedes utilizar un plugin como Brozzme DB Prefix[11].

Utiliza las *Security Keys* de WordPress

Cuando entras en tu panel de usuario de WordPress, o dejas un comentario, habitualmente se guardan ciertos datos en las *cookies*[12]. Si los datos no estuvieran encriptados podría aparecer algo como tu cuenta de correo o tu nombre de usuario, siendo algo que permitiría fácilmente detectar quién eres.

Desde la versión 2.6.0 de WordPress, existen unos pequeños algoritmos para encriptar esos datos y que sea más complejo saber quién eres o cómo acceder a tu usuario. Para ello has de configurar los siguientes elementos en tu fichero de configuración [wp-config.php]. Puedes crear tus propias claves, aunque lo mejor es que utilices una herramienta que genera códigos aleatorios y complejos desde la dirección del *Secret Key* de WordPress[13]. Esto hará que te aparezcan unos códigos similares a estos:

```
define('AUTH_KEY', 'l3Yk-= V+N@M&`=-skp,[F?Mp1vN|.tQ-mCQr-_YrUJ-');
define('SECURE_AUTH_KEY', 'u[G-;-XPjovJ_hy?v`IWUgf(/7mGy1R>Na.~Yld(jg~W');
define('LOGGED_IN_KEY', '-gQS^SH+{qCXb_=aBI=Q~x|aq@8`HU:Tt<XJ(j8KX>x*');
define('NONCE_KEY', 'iY]*URXkUJ5o=J0Q/b,P;%ULl1`v3x=>+#F]S|z&^<bz');
define('AUTH_SALT', 'FWlh^z8L;s4`*r>7H#*(iA!OWV9^X#^#m-A&>;!lz@(X');
define('SECURE_AUTH_SALT', 'oZ#3S6d{pjeTb.lxLy2uQec=Cs?oWRRm% *(U7!QFQ%(q');
define('LOGGED_IN_SALT', '=oMWAYx1UVXaZRK?slW}_q9Fbbjw7Bi|Ca|QLst^64/zF');
define('NONCE_SALT', '|)<Z~_/gf[.iiec-M/HM@|xW28LMIc%e<bn^og9+LVv1C');
```

El monstruo de las galletas (*cookies*)

Almacenar la información segura para saber quién está navegando por tu WordPress es importante y por eso lo es también saber dónde y cómo se guarda la información (y que esté segurizada).

[11] https://wordpress.org/plugins/brozzme-db-prefix-change/
[12] https://es.wikipedia.org/wiki/Cookie_(informática)
[13] https://api.wordpress.org/secret-key/1.1/salt/

Para ello, en una instalación simple de WordPress, definiremos concretamente dónde se almacenarán las galletas (*cookies*) para que no sean tan fácilmente accesibles.

```
define('COOKIE_DOMAIN', 'www.example.com');
define('COOKIEPATH', preg_replace('|https?://[^/]+|i', '', get_option('home').'/'));
define('SITECOOKIEPATH', preg_replace('|https?://[^/]+|i', '',
get_option('siteurl').'/'));
define('ADMIN_COOKIE_PATH', SITECOOKIEPATH.'wp-admin');
define('PLUGINS_COOKIE_PATH', preg_replace('|https?://[^/]+|i', '', WP_PLUGIN_URL));
```

Con esto las cookies se establecerán en el dominio principal que corresponda y posteriormente se irán guardando por carpetas según quién las necesite. Los usuarios del panel de administración sólo tendrán disponibles sus cookies cuando estén en el panel. Los *plugins* sólo tendrán acceso a sus *cookies*…

Un caso especial es de aquellas instalaciones con WordPress MultiSite[14] que, en ese caso, por la posibilidad de tener varios *hostname*, es mejor que el sistema configure todo de forma automática; lo mejor es no incluir ninguna de estas líneas de código o dejar sus contenidos vacíos y que lo gestione automáticamente el sistema.

Evita “manazas” y cambios de configuración

Uno de los errores más habituales de un usuario administrador en el panel de administración de WordPress es cambiar las direcciones URL que aparecen en la primera pantalla de configuración: las direcciones de la página principal del sitio.

Para evitar que cualquier persona, de forma deliberada o por error, haga un cambio en las direcciones y por tanto el sitio deje de funcionar, lo mejor es bloquear esas direcciones desde el fichero de configuración [wp-config.php]. Para ello añadiremos dos líneas:

```
define('WP_SITEURL', 'https://www.example.com');
define('WP_HOME', 'https://www.example.com');
```

Utiliza siempre un certificado TLS

Cuando navegamos por Internet queremos hacerlo de forma segura. Para ello deberíamos navegar por sitios de confianza “con la barrita verde” (o el candado). Este sistema diferencia el HTTP[15] [http://] del HTTPS[16] [https://]. A grandes rasgos la diferencia que hay entre uno y otro es que la información de usuarios, contraseñas, formularios, etcétera, en el segundo caso va encriptada desde tu navegador hasta el servidor. Así, si un *hacker* es capaz de interceptar tus datos, los vería encriptados y no podría conocer su contenido de forma fácil.

[14] https://codex.wordpress.org/Create_A_Network
[15] https://es.wikipedia.org/wiki/Protocolo_de_transferencia_de_hipertexto
[16] https://es.wikipedia.org/wiki/Protocolo_seguro_de_transferencia_de_hipertexto

Para conseguir esto hemos de instalar un certificado TLS[17] en el servidor web (Apache HTTPD, nginx…). Hay que tener en cuenta que habitualmente (mal) se llama a todos los certificados "SSL" (aunque esa tecnología es previa y obsoleta).

Es posible que por temas de compatibilidad a tus usuarios no les quieras ofrecer esto por defecto en todo el sitio, pero como mínimo tú si que deberías tenerlo configurado en el panel de administración [/wp-admin/].

Los pasos son los siguientes:

1. Consigue, instala y configura un certificado TLS. Puedes conseguir uno mediante el sistema abierto de Let's Encrypt[18] (por ejemplo, mediante el sitio web SSL for Free[19]) o comprando un certificado en cualquiera de los proveedores habituales.
2. En la medida de lo posible, configura todo tu sitio web para que únicamente funcione con HTTPS. En caso de no ser posible, al menos, que puedas navegar por HTTP y HTTPS indistintamente.
3. Asegúrate de que el certificado está funcionando correctamente. Existen herramientas[20] para verificarlos.
4. Cuando accedas al panel de administración, fuerza que se active el HTTPS.

Para hacer esto último, añade unas líneas en el fichero de configuración de WordPress [wp-config.php]:

```
define('FORCE_SSL_LOGIN', true);
define('FORCE_SSL_ADMIN', true);
```

Asegúrate de controlar tus usuarios

En WordPress los usuarios tienen un peso importante en el sistema. Tanto si tienes el gestor completamente cerrado y sólo accedes tú, como si tienes un centenar de personas trabajando con él, es importante evitar que quien no tenga que acceder, no lo haga.

Históricamente WordPress creaba como usuario por defecto el [admin]. Esto hace que las instalaciones más antiguas todavía puedan tenerlo y se debería eliminar, ya que es una fuente de intentos de acceso habituales (también lo son palabras como [root] o [administrador]). En general habría que evitar el uso de nombres de usuario simples y quizá es más recomendable que el nombre de usuario sea la cuenta de correo. Si tienes un usuario [admin] puedes crear otro usuario (administrador o no) y cuando vayas a borrarlo le transfieres todos los contenidos.

Otro asunto importante es el uso de contraseñas seguras. Una contraseña segura no es aquella que usa letras, números, mayúsculas, minúsculas, letras raras… pero sólo de 8 caracteres, sino una contraseña larga y fácil de recordar para el que la dispone. Es por eso que deberías obligar a tus usuarios a tener contraseñas de al menos 12 caracteres, aunque sea sólo alfanumérica.

Para acabar, un detalle importante es el de limitar los accesos de un usuario al panel. En realidad, el objetivo no es tanto limitar al usuario sino limitar a alguien

[17] https://es.wikipedia.org/wiki/Transport_Layer_Security
[18] https://letsencrypt.org/
[19] https://www.sslforfree.com/
[20] https://www.ssllabs.com/ssltest/

(persona, máquina…) a que haga muchos intentos de acceso ya sea con un usuario existente o probando distintos.

Aún todo esto, hoy en día existe una forma mucho más efectiva de mejorar la seguridad, y es configurar un sistema de doble verificación. Además de tener un usuario y una contraseña el objetivo es que, tras acceder correctamente, verifiques que realmente eres tú. ¿Cómo conseguirlo? Pues con algo que en general todos llevamos encima: nuestro teléfono móvil. La idea es instalar un *plugin* de doble verificación[21], obligatorio para todos los usuarios, de forma que una vez el usuario haya accedido con su usuario y contraseña (más o menos segura) te pedirá una nueva clave numérica que se genera cada minuto y que sólo estará configurada en tu dispositivo móvil. En la pantalla aparecerá ese número, lo introduces y ya podrás disfrutar de tu WordPress. Así, si alguien consigue tus accesos, no podrá acceder ya que también debería tener tu teléfono móvil. Incluso, existen sistemas aún más avanzados, como el Magic Password[22], en el que sólo necesitarás una App para Android o iOS.

Cambiar la numeración de los usuarios

Cuando se crean las tablas de WordPress, todos los sistemas de autoincremento comienzan en la cifra número 1. Esto hace que el primer usuario que crees será el usuario con identificador 1, el siguiente es el 2.

Los sistemas que atacan los usuarios de forma automática, aprovechan esta numeración y suelen analizar los usuarios entre el 1 y el 10, como punto de inicio. Es por esto que podríamos hacer un trabajo en este orden:

1. Creamos todo el sistema, y obviamente el primer usuario.
2. Ejecutamos la consulta siguiente (recuerda cambiar [wp_] por tu prefijo):

```
ALTER TABLE wp_users AUTO_INCREMENT = 512;
```

3. Creamos un nuevo usuario administrador en WordPress
4. Eliminamos el primer usuario que hemos creado.

Con este sistema, en principio, ese nuevo usuario (y los siguientes) habrán comenzado por la cifra 512, de forma que cualquier ataque que se quiera realizar de esta forma quedará invalidado.

Bloquear intentos de acceso masivos

Aunque una contraseña segura y doble verificación es suficiente para que nadie pueda entrar en tu WordPress mediante sistemas de fuerza bruta, puede ser un poco cansado que se estén ejecutando miles de solicitudes de acceso. Es por esto que podemos plantear la instalación de un *plugin* que cuando detecte este tipo de ataques los bloquee. Podemos instalar un Firewall (como se habla en la sección de Seguridad activa) o tenemos sistemas que se limitan a controlar los intentos de acceso de la zona de acceso[23].

A qué han de acceder tus usuarios

Por defecto WordPress lleva unos sencillos niveles de usuario[24]: Administrador, Editor, Suscriptor… Pero si comienzas a añadir funcionalidades que no son las sencillas de

[21] https://wordpress.org/plugins/2fas-light/
[22] https://wordpress.org/plugins/magic-password/
[23] https://wordpress.org/plugins/loginizer/
[24] https://codex.wordpress.org/Roles_and_Capabilities

publicación, tal vez sea mejor que tus usuarios tengan un rol muy concreto en el que sólo toquen lo que deban tocar.

Para ello existen varios plugins de gestión de usuarios, aunque te recomiendo el uso de User Role Editor[25] o de Members[26].

Cuando acabes de trabajar…

Es fácil: desconecta tu sesión. Dejar la sesión abierta en cualquier dispositivo ofrece una posibilidad de dejar las cookies y la sesión activa para que el que venga detrás tenga acceso a ella. Aunque sea un dispositivo personal, sal. No cuesta nada volver a poner tu contraseña.

Evita peticiones de servidores externos

WordPress es un gestor de contenidos que permite infinidad de opciones, entre ellas la de leer elementos externos (tan sencillos como *feeds* de noticias de otros sitios) o la descarga de otros elementos (como las plantillas y *plugins*). Este sistema viene activo por defecto lo que hace que cualquier elemento sea capaz de conectarse de forma externa.

Si se diera el caso de que se inyectase algún tipo de código malicioso, éste podría conectarse a otros sitios y descargar o actualizar información, algo que por norma general no deseamos que ocurra.

Si queremos bloquear los accesos externos podemos activar un elemento en el fichero de configuración:

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

Pero esto podría bloquear cualquier llamada externa, por lo que podemos activar una lista blanca de sitios a los que sí se podría acceder:

```
define('WP_ACCESSIBLE_HOSTS', '*.wordpress.org,*.github.com');
```

En este caso daríamos acceso a los sitios web de WordPress y Github.

Limpieza de imágenes y multimedia

Subir imágenes, editarlas, recortarlas y volver a empezar. Sin duda uno de los esfuerzos mayores en muchas versiones de WordPress ha sido la de trabajar en mejoras del sistema de edición de contenidos multimedia, principalmente las imágenes.

Cuando subimos una imagen (la original) automáticamente se crean otras ediciones más pequeñas de la misma que se usarán como destacados o en otros lugares de la plantilla. Pero posteriormente podemos editarlas y recortarlas. Esto genera otro *set* completo de imágenes sin sobrescribir las primeras.

[25] https://wordpress.org/plugins/user-role-editor/
[26] https://wordpress.org/plugins/members/

Si finalmente decides que estas imágenes han de desaparecer y eliminas la imagen original, no se eliminan todas estas imágenes recreadas que podrían contener datos que no te interesa que estén accesibles.

¿Cómo se pueden eliminar todas estas imágenes generadas cuando eliminas la imagen original? Pues activando un elemento en el fichero de configuración [wp-config.php]:

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

Ten en cuenta que si utilizas algún sistema del estilo a un CDN[27] es posible que estas imágenes ya se hayan replicado y distribuido por el mundo, por lo que tendrás que hacer un ejercicio de limpieza manual para eliminar esas imágenes que tal vez nunca debieron ver la luz.

Unifica los CSS y JavaScript

Aunque esta recomendación parezca más de Web Performance Optimization (WPO)[28], este sistema permite conseguir otro efecto colateral muy interesante para la seguridad de los sitios.

En general los ficheros de estilo (CSS) y los de *scripting* (JavaScript) incluyen cierta información como las versiones, ya sea en su nombre, en el interior, en algunos comentarios… el hecho de unificarlos, reducirlos y limpiarlos permite que internamente los datos sigan activos (necesarios para el buen funcionamiento de WordPress) pero invisibles para los usuarios.

Sistemas de reducción hay varios, normalmente en forma de plugin[29], y que básicamente han de permitir 3 opciones: limpiar, reducir y unificar CSS, limpiar, reducir y unificar JavaScript y limpiar, reducir y unificar el HTML.

El propio WordPress dispone de una forma similar que es la concatenación y compresión de elementos. Para ello habría que incluir en el fichero de configuración [wp-config.php] los siguientes elementos:

```
define('CONCATENATE_SCRIPTS', true);
define('COMPRESS_CSS', true);
define('COMPRESS_SCRIPTS', true);
```

Este sistema complica la detección automática, pero no reduce el riesgo por completo.

Usar un servicio externo de envío de correo

El correo basura (*spam*)[30], esa lacra de Internet que muchos sufrimos todos los días en nuestra bandeja de entrada, es algo que debemos evitar que ocurra debido a nuestro sitio web. Controlar el correo que sale desde nuestro servidor por norma general es algo que no se revisa ni a lo que se le presta atención, además de ser un correo

[27] https://es.wikipedia.org/wiki/Red_de_entrega_de_contenidos
[28] https://www.javiercasares.com/wpo/
[29] https://wordpress.org/plugins/fast-velocity-minify/
[30] https://es.wikipedia.org/wiki/Spam

saliente que no va firmado ni segurizado. Para mejorar y corregir esto, hacer que los correos lleguen correctamente y de forma segura y atractiva, lo mejor es utilizar un servicio de salida de correo externo (SMTP)[31].

Una vez más, existen decenas de *plugins* para sustituir las funciones `mail()` internas por externas[32], aunque en este caso recomiendo utilizar Elastic Email[33] que además tiene un *plugin* propio[34] que facilita su configuración.

Ocultar cabeceras inconvenientes

Como la mayor parte de gestores de contenidos, WordPress se identifica claramente y ofrece determinados servicios que luego usaremos o no. Y habitualmente este hecho implica pequeños elementos que te dejan ver información y disminuyen la seguridad.

Es por esto que es muy recomendable ocultar determinadas cabeceras que aparecen en el `<head>` (u otros elementos, como los *feeds*) de la página.

Existen un par de decenas de códigos que se pueden eliminar si no queremos tener determinados elementos funcionando:

```
remove_action('set_comment_cookies', 'wp_set_comment_cookies');
add_filter('show_admin_bar','__return_false');
add_filter('the_generator', '__return_false');
remove_action('wp_head', 'adjacent_posts_rel_link', 10, 0);
remove_action('wp_head', 'adjacent_posts_rel_link_wp_head', 10, 0);
remove_action('wp_head', 'feed_links', 2);
remove_action('wp_head', 'feed_links_extra', 3);
//remove_action('wp_head', 'index_rel_link');
//remove_action('wp_head', 'parent_post_rel_link', 10, 0);
remove_action('wp_head', 'rsd_link');
//remove_action('wp_head', 'start_post_rel_link', 10, 0);
remove_action('wp_head', 'wlwmanifest_link');
remove_action('wp_head', 'wp_generator');
remove_action('wp_head', 'wp_shortlink_wp_head', 10, 0);
```

Te recomiendo que si quieres saber qué es cada función, lo analices con la documentación del Codex de WordPress[35]. Las acciones comentadas lo están porque son de versiones antiguas, en principio ya obsoletas.

Una de las posibilidades es la de obligar a eliminar todos estos elementos de forma automática mediante la creación de un elemento añadido (*plugin*) obligatorio. Para ello subiremos -por FTP, SSH- el plugin [WPdanger Headers][36] a la carpeta [/wp-content/mu-plugins/]. Este plugin se activará siempre en el sitio y eliminará todo de forma automática. Es muy recomendable e importante que cambies o comentes las líneas que no te interesen.

---

31 https://www.casares.blog/proveedores-smtp/
32 https://wordpress.org/plugins/wp-smtp/
33 https://elasticemail.com/
34 https://wordpress.org/plugins/elastic-email-sender/
35 https://developer.wordpress.org/reference/
36 http://bit.ly/wpdanger-headers

Controla a los robots[.txt]

El fichero de robots.txt que ha de estar en la carpeta raíz del *hostname* de la instalación (aunque tu WordPress esté en una carpeta), es un fichero que no viene por defecto en WordPress, aunque si lo llamas el sistema genera uno sencillo, y que deberías crear.

Este fichero de texto[37] lo que hace es decirle a los robots de búsqueda (como GoogleBot, BingBot, YandexBot, Slurp!...) qué deben y qué no deben analizar de tu sitio.

¿Qué deberíamos bloquear de WordPress en este fichero? En principio nada. Hoy en día se pueden conseguir bloquear muchos elementos mediante las configuraciones internas de plugins de seguridad, por lo que si quieres tener un sitio seguro lo mejor es no dar información en este fichero.

De esta manera, el fichero tendrá un contenido simple:

```
User-Agent: *
```

Evitar spam en comentarios

Un ataque habitual en los sitios con WordPress (principalmente en aquellos que están focalizados en escribir entradas tipo *blog*) es el que se hace de los comentarios mediante ataques por spam. Para ello es básico tener instalado un sistema *antispam* en los comentarios.

WordPress suele venir con uno instalado (no activado) de serie, que es Akismet[38]. No es el único que existe, todo lo contrario, la lista es bastante amplia y larga[39].

Analiza tus enlaces

La seguridad no sólo consiste en que tú te protejas sino también en proteger a los demás. Es por esta razón que uno de los elementos que deberías revisar cada cierto tiempo son los enlaces salientes de tu sitio web. En muchas ocasiones los enlaces dejan de funcionar o se redirigen a sitios de, digámoslo finamente, dudosa calidad.

Es por eso que es muy recomendable utilizar algún complemento que analice los enlaces rotos[40].

Pero no sólo los enlaces que tú publicas, sino aquellos que te hacen referencia. Los *trackbacks* (sitios webs que te enlazan y aparecen como comentarios) también pueden falsificarse simplemente para hacer spam o para llevar a tus usuarios a sitios poco seguros. Para estos casos puedes desactivarlos, o verificar que cuando se produce uno, realmente te han enlazado desde allí[41].

[37] http://www.robotstxt.org/
[38] https://wordpress.org/plugins/akismet/
[39] https://wordpress.org/plugins/tags/spam/
[40] https://wordpress.org/plugins/wp-link-status/
[41] https://wordpress.org/plugins/simple-trackback-validation-with-topsy-blocker/

Evitar ataques mediante XML-RPC

Una de las ventajas de WordPress es su flexibilidad a la hora de ser utilizado por aplicaciones de terceros, y para ellos muchas utilizan el estándar XML-RPC[42] que permite la interacción con el número del gestor de contenidos.

Obviamente, si desactivas esta tecnología no podrás usar programas como Open Live Writer[43] o herramientas como IFTTT[44] e incluso la propia App de WordPress para Android[45] o iOS[46].

Para mejorar la seguridad, una de las posibilidades es la de obligar a bloquearlo todo de forma automática mediante la creación de un elemento añadido (*plugin*) obligatorio. Para ello subiremos -por FTP, SSH- el plugin [WPdanger XML-RPC][47] a la carpeta [/wp-content/mu-plugins/]. Este plugin se activará siempre en el sitio y bloqueará todo de forma automática.

Como este método puede ser muy agresivo, te puedes plantear otras opciones más ligeras que en un futuro cercano te permitan añadir, por ejemplo, una IP desde la que tú puedas acceder, pero el resto no.

En Apache HTTPD (dentro del fichero .htaccess):

```
<Files xmlrpc.php>
  order deny, allow
  deny from all
  allow from 8.8.8.8
</Files>
```

En nginx (dentro del fichero de configuración del sitio):

```
location = /xmlrpc.php {
  limit_except POST {
    deny all;
  }
  allow 8.8.8.8;
  access_log off;
  log_not_found off;
}
```

Existe una herramienta muy interesante para verificar el funcionamiento o no de esta tecnología, llamada WordPress XML-RPC Validation Service[48].

[42] https://es.wikipedia.org/wiki/XML-RPC
[43] http://openlivewriter.org/
[44] https://ifttt.com/
[45] https://play.google.com/store/apps/details?id=org.wordpress.android
[46] https://itunes.apple.com/app/wordpress/id335703880?mt=8
[47] http://bit.ly/wpdanger-xmlrpc
[48] http://xmlrpc.eritreo.it/

Copias de seguridad

Si hablamos de seguridad hemos de hablar de copias de seguridad (*backups*). No tiene sentido poner medidas que ayudan a la seguridad de tu sitio si luego no tienes una copia del mismo que permita restaurar una configuración fallida.

Cuando hablamos de copias de seguridad con WordPress hablamos de dos (o tres) partes, dependiendo de tu sitio. Las dos principales son la base de datos y los ficheros. La tercera es la del certificado TLS que permite el acceso por HTTPS.

Las copias de seguridad se pueden realizar de muchas maneras, manualmente (como haría un buen administrador de sistemas 😉) o de forma automática mediante un *plugin*. Una vez más, para esto hay decenas de ellos[49], aunque te recomiendo uno en concreto[50] que permite almacenar los datos generados en varios lugares.

Además de tener copias de seguridad es importante dejar una de ellas en la propia máquina (no accesible por web) y mandar otra a cualquier otro lugar (Dropbox, Amazon, un servidor externo…).

Finalmente, tendremos una copia de seguridad de los ficheros del certificado TLS. Estos ficheros no son necesario que se copien cada día o semana, ya que se actualizan como pronto cada 3 meses.

Y finalmente la pregunta del millón: ¿cada cuánto he de hacer las copias de seguridad? Pues en general dependerá de cuánto cambia tu sitio con WordPress. Si cada día publicas varias entradas o contenidos, es posible que sea interesante realizar una copia cada 6-12 horas. Si publicas o cambias elementos una vez cada mucho, con una copia semanal puede que ya tengas suficiente.

Seguridad activa

Y si hacer copias de seguridad podríamos considerarlo seguridad pasiva, también podemos activar determinadas herramientas de seguridad activa que intenten evitar "lo más en tiempo real posible" ataques o cambios.

Para llegar a este nivel podemos considerar dos frentes a tener en cuenta: bloquear ataques (*firewall*) y bloquear cambios de ficheros (*malware*).

Como en casi todos los casos, existen muchos plugins que realizan estas tareas, aunque también podemos mirar si nuestro proveedor de alojamiento web (*hosting*) ofrece uno externo. En cualquier caso, recomiendo el uso de Wordfence Security[51] que tiene muchas configuraciones en su opción *freemium*. Obviamente, si recibes un nivel de ataques muy elevado te recomiendo algo más pro (ya sea con este mismo plugin o con otra solución).

Para analizar posibles ataques en los ficheros o en el mismo núcleo de WordPress deberíamos de verificar fichero a fichero que estos no han cambiado. Para conseguirlo existen soluciones como Sucuri Security[52] que buscan diferencias entre los ficheros originales del núcleo, plantillas, *plugins*… y si hay diferencias te avisa.

[49] https://wordpress.org/plugins/tags/backup/
[50] https://wordpress.org/plugins/backwpup/
[51] https://wordpress.org/plugins/wordfence/
[52] https://wordpress.org/plugins/sucuri-scanner/

Herramientas para Webmasters

Tener un WordPress es tener un sitio web, y como tal te convierte en *webmaster*. Esto significa que por Internet te encontrarás muchas herramientas que podrás utilizar para analizar tu sitio web. No voy a poner la infinidad de herramientas, pero sí algunas importantes que, entre otras cosas, te pueden ayudar a encontrar detalles de la seguridad en tu WordPress.

Baidu 百度站长平台
https://zhanzhang.baidu.com/

Bing Webmaster Tools
https://www.bing.com/toolbox/webmaster

DNSBL
http://www.dnsbl.info/

Google Search Console
https://www.google.com/webmasters/

REDbot
https://redbot.org/

Yandex Webmaster
https://webmaster.yandex.com/

Activar cachés

Una vez más, lo que probablemente sea una recomendación general y muy útil, como la de activar las cachés[53] de WordPress, se convierte en un elemento que ayuda a aumentar la seguridad de tu sitio. Porque cuando hablamos de seguridad no sólo hablamos de *hackeos*, sino de posibles ataques DDoS[54] que lo que hacen es dejar inaccesible tu sitio y que tus usuarios no puedan visitarlo.

WordPress tiene varios tipos de niveles de caché, pero independientemente de la que usemos lo primero es activarlo en el fichero de configuración [wp-config.php]:

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

A partir de este momento el sistema ya gestiona la caché de forma automática. Quizá no es la configuración óptima, pero funciona. Así que lo siguiente es gestionar esta caché y para ello, obviamente, usaremos cualquiera de los plugins de cache[55] que existen. Uno sencillo y funcional (además de completo) es el Super Cache[56].

Y como decía al principio, cachés en WordPress hay de varios tipos, y generalmente se habla de la de las páginas, pero existen otros niveles como los objetos. Para ello es

[53] https://es.wikipedia.org/wiki/Caché_web
[54] https://es.wikipedia.org/wiki/Ataque_de_denegación_de_servicio
[55] https://wordpress.org/plugins/tags/cache/
[56] https://wordpress.org/plugins/wp-super-cache/

muy recomendable utilizar algún sistema como memcached[57] o Redis[58] para su almacenamiento.

La opción de Cloudflare

A la hora de escalar, cachear, evitar ataques nunca hay una situación ideal, pero sí servicios por Internet que te ayudan a protegerte de una forma más o menos sencilla. Una de estas opciones es la de Cloudflare[59], que básicamente consiste en activar un servicio en el que ponen una capa intermedia entre el usuario y tu sitio web.

Una vez tengas cuenta con ellos, lo primero es activar su plugin[60] en tu sitio. A partir de aquí podrás comenzar a jugar con algunas opciones. Por norma general este servicio no te cacheará las páginas, pero sí que lo hará con otros elementos estáticos. Ejercerá básicamente de CDN (*Content Delivery Network*), lo que significa que tus imágenes, estilos y scripts se distribuirán para una descarga más rápida. Esto lleva una contra, y es que en caso de que tú hagas cambios, también tendrás que avisar a Cloudflare de ello (en principio, para eso está el plugin).

Existen varios manuales que te pueden ser de ayuda:

- Using Cloudflare with WordPress
- Hardening WordPress Security
- Caching Static HTML with WordPress/WooCommerce
- Speed Up WordPress and Improve Performance

Recuerda que Cloudflare no evitará que pueda haber ataques a tu sitio, pero sí que ayudará a complicarlo un poco debido a que están entre los usuarios / atacantes y tu servidor real.

Elimina copias antiguas de entradas

Una vez más volvemos a temas que afectan principalmente a rendimiento y, en este caso, es el de ir vaciando la base de datos de copias antiguas de entradas o páginas. Al igual que comentaba con las imágenes, según vas actualizando contenidos se van creando copias antiguas. Por norma general esas copias no están accesibles, pero si quieres eliminar algún elemento que puede tener contenidos que nunca debieron estar ahí, lo mejor es limitar el tiempo o cantidad de elementos anteriores.

Para ello añadiremos al fichero de configuración [wp-config.php] el número de días que queremos que un elemento siga en la papelera:

```
define('EMPTY_TRASH_DAYS', 1);
```

El siguiente elemento es si queremos hacer copias de seguridad de las entradas, cuántas, si sí:

[57] https://memcached.org/
[58] https://redis.io/
[59] https://www.cloudflare.com/
[60] https://wordpress.org/plugins/cloudflare/

```
define('WP_POST_REVISIONS', 5);
```

o si no:

```
define('WP_POST_REVISIONS', false);
```

Gestionar licencias de plantillas o plugins

Existen servicios de pago que requieren una licencia para funcionar y habitualmente estas licencias se añaden desde el panel de administración. Esto hace que la licencia esté visible para todos aquellos que tienen permisos y que permitiría que algún "listillo" la pueda intentar copiar y usar donde no debe.

Para evitarlo, algunos de estos plugins o plantillas permiten configurar la licencia en el fichero de configuración, de forma que quede fuera del alcance de quienes no deben tenerla.

Un ejemplo es el de Gravity Forms[61] que permitiría añadir una línea en el fichero de configuración [wp-config.php] similar a la siguiente:

```
define('GF_LICENSE_KEY', 'abcde123456789');
```

Otro ejemplo más habitual puede ser el de WordPress.com, que suele ser necesario para plugins como JetPack[62] o Akismet[63]:

```
define('WPCOM_API_KEY', 'abcde123456789');
```

Plantilla por defecto en caso de error

Aunque normalmente no es algo que suela pasar, es posible que tu plantilla falle por alguna razón (por alguna incompatibilidad, porque se borren algunos ficheros…) o simplemente que tengas un WordPress MultiSite y quieras indicar cuál de tus plantillas es la que se configurará con los nuevos sitios, hay que añadir esta línea en el fichero de configuración [wp-config.php]:

```
define('WP_DEFAULT_THEME', 'twentysixteen');
```

En general te recomiendo que siempre tengas una de las plantillas propias de WordPress como seguridad, por ejemplo, la TwentySixteen[64].

[61] http://www.gravityforms.com/
[62] https://wordpress.org/plugins/jetpack/
[63] https://wordpress.org/plugins/akismet/
[64] https://wordpress.org/themes/twentysixteen/

Subir ficheros de cualquier tipo

Cuando hablamos de subir elementos multimedia a través del panel de administración solemos hacer referencia a imágenes, documentos y ficheros conocidos.

Esta propuesta en realidad va en contra de la seguridad, ya que lo que permite es todo lo contrario, que los administradores del sitio puedan subir cualquier tipo de fichero a través del panel; aun así, por determinadas necesidades, es posible que lo requieras activar en el fichero de configuración [wp-config.php]:

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

Conviértete en el Gran Hermano

Si tienes esa sensación de que pasa algo, alguien está tocando algo o no sabes bien qué y, además te gusta ser un poco el gran hermano, lo que te recomiendo es que instales un sistema de auditoría de seguridad, que te explique y detalle todo lo que va ocurriendo en tu WordPress a la mayor cantidad de niveles posibles.

Para ello puedes ir analizando los logs de tu servidor web, los de la base de datos, del servidor, etcétera, aunque también puedes instalar un complemento como el WP Security Audit Log[65] que generará una lista de documentación de todo lo ocurrido en tu sitio, de forma que puedas analizar y auditar si ha ocurrido algo indebido.

GDPR (General Data Protection Regulation)

A partir de finales de mayo de 2018 las empresas europeas, que trabajen en Europa o aquellos usuarios que lo hagan, van a estar sometidos a la nueva regulación de protección de datos aprobada a mediados de 2016. Esta nueva regulación, que sustituye la de 1995 y que viene dada principalmente por el terrorismo y todos los cambios digitales de estas dos últimas décadas, va a ser aplicada "tal cual", sin que los países tengan que adaptarla a su legislación propia.

Esta nueva legislación afecta, por ejemplo, a todas aquellas empresas exteriores a la UE que operen con usuarios europeos, además de estandarizar la legislación de datos a todos los países miembros. El incumplimiento de esta legislación puede alcanzar un 4%-5% de la facturación mundial de las compañías que la incumplan.

Entre otros detalles, esta legislación vendría a decirte que has de recoger información sobre algunos de los elementos que WordPress proporciona en tu sitio, como por ejemplo el registro de usuarios, comentarios, datos de contactos de los formularios, datos sobre analítica, etcétera. Sin duda, el plugin de WP Security Audit Log[66] comentado anteriormente puede ser muy útil en este caso.

Otro detalle a tener presente es que si tu sitio tiene una brecha de seguridad (alguien accede donde no debe) hay un límite de 72 para avisar a la Autoridad Supervisora. En este caso es muy recomendable usar algún complemento de cortafuegos que te avise en tiempo real de intentos de acceso o de ellos mismos.

[65] https://wordpress.org/plugins/wp-security-audit-log/
[66] https://wordpress.org/plugins/wp-security-audit-log/

También has de informar de cómo se almacenarán los datos de los usuarios (y cómo tendrán acceso a ellos), su derecho al olvido (cómo vas a eliminar sus datos) y la portabilidad de sus datos (en caso de que quieran llevar su material a otro sitio).

Otro detalle importante es el de los plugins, ya que aquellos que muevan datos de los usuarios fuera del sitio (un ejemplo conocido podría ser el de Jetpack) han de tener una política clara de cumplimiento de la GDPR.

Si estás interesado en este asunto, te recomiendo leer un artículo sencillo (no soy abogado, por lo que son simples conclusiones personales) que escribí sobre esta legislación en Ley de protección de datos europea.

Algunas preguntas, no sé si frecuentes

P- ¿Por qué he de hacerle caso a este documento y no a lo que pone en Internet?

R- Los manuales que hay por Internet están bien, pero la mayoría se basan en lo que ponen otros documentos sin plantearse el porqué de las cosas. Un ejemplo sencillo es que todos te hablan de borrar el usuario [admin], cuando hace años que por defecto ese usuario no existe.

P- ¿Por qué no has mencionado en ningún momento bloquear el spam de los comentarios según el [referrer]?

R- Pues porque hoy en día casi todos los sitios utilizan HTTPS, y eso implica que la cabecera de *referrer* suela ir vacía. Si aplicamos esas reglas, bloquearías casi todo.

P- Si tengo alguna duda, ¿puedo escribirte?

R- Sí, por supuesto; de todas formas, puede que tarde un poco en contestarte, o que mi respuesta sea que hables con alguien o busques respuesta en algún lugar concreto de la comunidad. Si te contesto con un enlace que te da la respuesta, pero no digo nada más… así soy yo (simple, como un botijo).

Sentido Común

Dicen que el sentido común es el menos común de los sentidos, y por eso te pido un poco de él. La seguridad en general es compleja, no hay nada 100% seguro, y por eso hay que aplicar todos estos consejos y códigos con cabeza, sabiendo el porqué los vas a aplicar y no simplemente copiando y pegando líneas de código "porque sí".

Como sabes, WordPress es seguro, y este documento lo que pretende es poner trabas a aquellos que quieran atacar tu sitio por la razón que sea. Simplemente eso.

Sobre el autor

Me llamo **Javier Casares** y llevo usando **WordPress** desde la versión 1.5 (allá por 2005) *¡Hola Javieeer!*

Colaboro con la comunidad de WordPress Barcelona y soy el creador del sitio WPdanger que analiza algunos de los elementos de seguridad de los sitios web creados con WordPress.

Si quieres saber de mi tan sólo has de visitar mi sitio web javiercasares.com donde tienes muchas formas de ponerte en contacto conmigo y de saber qué hago en la vida.

javier@casares.org

@JavierCasares

Checklist inicial

- [x] Leer WPdanger: Guía de Seguridad para WordPress
- [ ] Haz una copia de seguridad
- [ ] Actualiza WordPress
- [ ] Elimina plugins y plantillas que no utilices
- [ ] Actualiza plugins, plantillas y traducciones
- [ ] Revisa que todos los usuarios que tienes, son los que han de estar
- [ ] Instala un complemento de “2FA” (doble verificación de acceso)
- [ ] Revisa los permisos en los ficheros del sistema
- [ ] Instala un certificado TLS si no usas HTTPS
- [ ] Activa las herramientas para Webmasters
- [ ] Activa la caché
- [ ] Haz una copia de seguridad (sí, otra vez)