Puesta a punto

Todo lo que te surga para completar, añadir, modificar tu instalación siguiendo las directrices de Directadmin

- Cómo compilar una extensión o módulo de PHP en DirectAdmin
- Configuración de Headers de Seguridad en Nginx con DirectAdmin
- DMARC masivo — da_dmarc_bulk.sh

Cómo compilar una extensión o módulo de PHP en DirectAdmin

Introducción

Una buena práctica que sigo en mis proyectos de Laravel, es la de añadir en el `composer.json` del proyecto todo aquello que necesito, inclusive las extensiones, porque no todos los paneles, no todos los SO o no todas las distribuciones, es decir, no todas las instalaciones de de un servidor o equipo de desarrollo, tiene los mismos componentes por defecto.

En el caso de hoy al migrar un despligue de Laravel de cPanel a un DirectAdmin, cuando fui ha hacer un `composer update` salto un error indicándome la necesidad de tener la extensión `pcntl`.

Gracias a las buenas prácticas y los principios SOLID de programación, es fácil evitar un disgusto de proque no funciona, y donde esta el fallo.

Composer.json

```
{
    ...
    "require": {
        "php": "^8.1",
        "ext-pdo": "*",
        "ext-redis": "*",
        "ext-zlib": "*",
        "ext-pcntl": "*",
        ...
```

Como instalar la extensión en PHP en un servidor Directadmin

DirectAdmin tiene u número determinado de extensiones que desde la administración se pueden incluir en todas las instalaciones o desactivarlas. Pero también tiene u mecanismo sencillo para incorporar módulos de PHP a nuestra instalación, definidos en su documentación, Add a custom

module to PHP Cuastombuild

Localizar las configuraciones de PHP

```
da build used_configs | grep 'configure\.php'
PHP 8.3 configuration file: /usr/local/directadmin/custombuild/custom/php/configure.php83
PHP 8.2 configuration file: /usr/local/directadmin/custombuild/custom/php/configure.php82
PHP 5.6 configuration file: /usr/local/directadmin/custombuild/configure/php/configure.php56
PHP 7.4 configuration file: /usr/local/directadmin/custombuild/configure/php/configure.php74
```

Usar el override de Directadmin

```
VERSION=82
cd /usr/local/directadmin/custombuild
mkdir -p custom/php
cp -fp "configure/php/configure.php$VERSION" "custom/php/configure.php$VERSION"
```

Después deberás añadir el código necesario en el fichero `custom/php/configure.php$VERSION`, para que en la compilación de PHP se añade el módulo.

Verificamos:

```
cat  configuration file: /usr/local/directadmin/custombuild/custom/php/configure.php82
...
--enable-mbstring \
--enable-intl \
   --enable-pcntl
```

Puedes usar esto mismo para usarlos en tantas versiones como desees.

> Ahora DirectAdmin ya soporta hasta 8 versiones de PHP pero recuerda, que no es buena idea añadirlo a todo por muchas razones. Carga innecesaria, versiones antiguas que igual no lo soporte, y forzar al usuario a que actualice su versión a las actuales.

Completar el trabajo

Compilar

```
da build php
```

Reiniciar

```
systemctl restart httpd
systemctl restart nginx // Si no usas httpd
systemctl restart php-fpm$VERSION
```

Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

Configuración de Headers de Seguridad en Nginx con DirectAdmin

Introducción

Esta guía documenta cómo configurar headers de seguridad en servidores con DirectAdmin, tanto a nivel global como para dominios específicos que requieran exclusiones.

Última actualización: 25/12/2024 GMT+1

Configuración Global

La configuración global se aplica a **todos los dominios** del servidor mediante el archivo:

```
/usr/local/directadmin/data/templates/custom/cust_nginx.CUSTOM.post
```

Crear el directorio custom (si no existe)

```
mkdir -p /usr/local/directadmin/data/templates/custom
```

Ejemplo de configuración global

```
# Security Headers - Global
add_header X-Content-Type-Options "nosniff" always;
add_header X-Frame-Options "SAMEORIGIN" always;
add_header X-XSS-Protection "1; mode=block" always;
add_header Referrer-Policy "strict-origin-when-cross-origin" always;
add_header Permissions-Policy "geolocation=(), microphone=(), camera=()" always;
add_header Strict-Transport-Security "max-age=31536000; includeSubDomains" always;
```

```
add_header Content-Security-Policy "default-src 'self'; script-src 'self' 'unsafe-inline' 'unsafe-eval' https:; style-src 'self' 'unsafe-inline' https:; img-src 'self' data: https:; font-src 'self' data: https:; connect-src 'self' https:; frame-src 'self' https://www.youtube.com https://www.google.com; frame-ancestors 'self'; base-uri 'self'; form-action 'self';" always;

# Custom error pages for nginx-only domains
error_page 400 /error/400.html;
error_page 401 /error/401.html;
error_page 403 /error/403.html;
error_page 404 /error/404.html;
error_page 405 /error/405.html;
error_page 406 /error/406.html;
error_page 413 /error/413.html;
error_page 422 /error/422.html;
error_page 429 /error/429.html;
error_page 431 /error/431.html;
error_page 500 /error/500.html;
error_page 502 /error/502.html;
error_page 503 /error/503.html;
error_page 504 /error/504.html;

location ^~ /error/ {
    alias |DOCROOT|/error/;
    internal;
    # Repetir headers en este location block
    add_header X-Content-Type-Options "nosniff" always;
    add_header X-Frame-Options "SAMEORIGIN" always;
    add_header X-XSS-Protection "1; mode=block" always;
    add_header Referrer-Policy "strict-origin-when-cross-origin" always;
    add_header Permissions-Policy "geolocation=(), microphone=(), camera=()" always;
    add_header Strict-Transport-Security "max-age=31536000; includeSubDomains" always;
    add_header Content-Security-Policy "default-src 'self'; script-src 'self' 'unsafe-inline' 'unsafe-eval' https:; style-src 'self' 'unsafe-inline' https:; img-src 'self' data: https:; font-src 'self' data: https:; connect-src 'self' https:; frame-src 'self' https://www.youtube.com https://www.google.com; frame-ancestors 'self'; base-uri 'self'; form-action 'self';" always;
}
```

Aplicar cambios

Después de crear o modificar el archivo, reconstruir las configuraciones de nginx:

```
cd /usr/local/directadmin/custombuild
./build rewrite_confs
```

Personalización por Dominio

Para dominios que requieren configuraciones diferentes (CSP más permisivo, exclusiones de headers, etc.), DirectAdmin permite crear archivos de personalización por dominio.

Ubicación de archivos

```
/usr/local/directadmin/data/users/<USUARIO>/domains/<DOMINIO>.cust_nginx
/usr/local/directadmin/data/users/<USUARIO>/domains/<DOMINIO>.cust_nginx.2
/usr/local/directadmin/data/users/<USUARIO>/domains/<DOMINIO>.cust_nginx.3
/usr/local/directadmin/data/users/<USUARIO>/domains/<DOMINIO>.cust_nginx.4
```

Archivo de configuración personalizada en home del usuario

Para configuraciones más complejas, se puede crear un archivo `.conf` en el directorio del usuario e incluirlo:

```
/home/<USUARIO>/nginx/<nombre>.conf
```

Puntos de Inserción en DirectAdmin

DirectAdmin utiliza tokens en la plantilla `nginx_server.conf` para insertar configuraciones personalizadas:

Token	Archivo por Dominio	Ubicación en nginx

`CUSTOM1`	`.cust_nginx`	Antes del bloque `server {}`
`CUSTOM`	(interno)	Inicio del bloque `server {}`
`CUSTOM2`	`.cust_nginx.2`	Dentro de `location / {}`
`CUSTOM3`	`.cust_nginx.3`	Después de locations, antes de webapps
`CUSTOM4`	`.cust_nginx.4`	Final del bloque `server {}`

Archivo .cust_nginx (CUSTOM1)

Se usa principalmente para:

- Cambiar el DOCROOT (aplicaciones Laravel, etc.)
- Configuraciones que van antes del bloque server

Ejemplo - Cambiar DOCROOT para Laravel:

```
|?DOCROOT=`HOME`/domains/`DOMAIN`/mi-app/public|
```

Archivo .cust_nginx.3 (CUSTOM3)

Se usa para:

- Configuraciones de cache
- Location blocks adicionales
- Headers específicos para tipos de archivo

Ejemplo - Cache de assets:

```
location ~* \.(js|css|png|jpg|jpeg|gif|svg|ico)$ {
    expires 30d;
    add_header Cache-Control "public, no-transform";
}
location ~* \.(jpg|jpeg|gif|png|svg)$ {
    expires 365d;
}
location ~* \.(pdf|css|html|js|swf)$ {
    expires 2d;
}
```

Archivo .cust_nginx.4 (CUSTOM4)

Se usa para:

- Incluir archivos de configuración externos
- Location blocks complejos
- Reglas de rewrite específicas

Ejemplo - Incluir configuración WHMCS:

```
include /home/intranet/nginx/whmcs.conf;
```

Ejemplos Prácticos

Ejemplo 1: Dominio con CSP relajado

Para un dominio que necesita permitir scripts de terceros (ej: pasarelas de pago, widgets):

Archivo: `/usr/local/directadmin/data/users/usuario/domains/tienda.com.cust_nginx.4`

```
# Sobrescribir CSP global con uno más permisivo
add_header Content-Security-Policy "default-src 'self'; script-src 'self' 'unsafe-inline'
'unsafe-eval' https: blob:; style-src 'self' 'unsafe-inline' https:; img-src 'self' data:
https: blob:; font-src 'self' data: https:; connect-src 'self' https: wss:; frame-src 'self'
https:; frame-ancestors 'self'; base-uri 'self'; form-action 'self' https:;" always;
```

Ejemplo 2: Dominio que permite iframes externos

Para un dominio que necesita ser embebido en otros sitios:

Archivo: `/usr/local/directadmin/data/users/usuario/domains/widget.com.cust_nginx.4`

```
# Permitir que el sitio sea embebido
add_header X-Frame-Options "" always;
add_header Content-Security-Policy "default-src 'self'; frame-ancestors https://sitio-
```

```
padre.com https://otro-sitio.com;" always;
```

Ejemplo 3: Aplicación Laravel con DOCROOT personalizado

Archivo: `/usr/local/directadmin/data/users/develop/domains/larafactu.com.cust_nginx`

```
|?DOCROOT=`HOME`/domains/`DOMAIN`/larafactu/public|
```

Ejemplo 4: BookStack Wiki

Archivo: `/usr/local/directadmin/data/users/castris/domains/wiki.castris.com.cust_nginx`

```
|?DOCROOT=`HOME`/domains/`DOMAIN`/BookStack/public|
```

Ejemplo 5: Configuración compleja con archivo externo

Crear archivo de configuración en home del usuario:

Archivo: `/home/intranet/nginx/whmcs.conf`

```
# WHMCS CONFIG
location ~ /announcements/?(.*)$ {
    rewrite ^/(.*)$ /index.php?rp=/announcements/$1;
}

location ~ /download/?(.*)$ {
    rewrite ^/(.*)$ /index.php?rp=/download$1;
}

location ~ /knowledgebase/?(.*)$ {
    rewrite ^/(.*)$ /index.php?rp=/knowledgebase/$1;
}

# ... más reglas de rewrite ...
```

```
# Security Advisory
location ^~ /vendor/ {
    deny all;
    return 403;
}
```

Archivo: `/usr/local/directadmin/data/users/intranet/domains/intranet.castris.com.cust_nginx.4`

```
include /home/intranet/nginx/whmcs.conf;
```

Ejemplo 6: Deshabilitar todos los headers de seguridad (NO RECOMENDADO)

Solo para debugging o situaciones muy específicas:

Archivo: `/usr/local/directadmin/data/users/usuario/domains/debug.com.cust_nginx.4`

```
# TEMPORAL - Solo para debugging
add_header X-Content-Type-Options "" always;
add_header X-Frame-Options "" always;
add_header X-XSS-Protection "" always;
add_header Referrer-Policy "" always;
add_header Permissions-Policy "" always;
add_header Strict-Transport-Security "" always;
add_header Content-Security-Policy "" always;
```

Troubleshooting

Verificar sintaxis de nginx

```
nginx -t
```

Ver configuración generada para un dominio

```
cat /etc/nginx/conf.d/usuario.dominio.com.conf
```

Reconstruir todas las configuraciones

```
cd /usr/local/directadmin/custombuild
./build rewrite_confs
```

Recargar nginx sin reiniciar

```
systemctl reload nginx
```

Verificar headers de un sitio

```
curl -I https://dominio.com
```

Logs de errores

```
tail -f /var/log/nginx/domains/dominio.com.error.log
```

Notas importantes

1. **Orden de precedencia**: Los headers definidos en location blocks más específicos sobrescriben los globales.
2. **Keyword always**: Usar siempre `always` al final del `add_header` para que se aplique en todas las respuestas (incluyendo errores).
3. **Reconstruir después de cambios**: Siempre ejecutar `./build rewrite_confs` después de modificar archivos `.cust_nginx*`.
4. **Permisos**: Los archivos deben tener permisos correctos para que DirectAdmin pueda leerlos.
5. **Variables de DirectAdmin**: Se pueden usar variables como `|DOCROOT|`, `|DOMAIN|`, `|HOME|` en las configuraciones.

Variables Disponibles en Templates

Variable	Descripción
`|DOMAIN|`	Nombre del dominio
`|HOME|`	Directorio home del usuario
`|DOCROOT|`	Document root del dominio
`|IP|`	IP del servidor
`|PORT_80|`	Puerto HTTP
`|PORT_443|`	Puerto HTTPS

Referencias

- DirectAdmin Custom Nginx Templates
- Mozilla Observatory - Para verificar headers de seguridad
- Security Headers - Análisis de headers

Aviso

Esta documentación se entrega tal y como está, basada en configuración del servidor dar.tabratino.com.

DMARC masivo — da_dmarc_bulk.sh

Descripción

Script para crear registros `_dmarc` TXT en todos los dominios de un servidor DirectAdmin que no lo tengan. Usa la API local de DA (`CMD_API_DNS_CONTROL`), que propaga automáticamente los cambios a los miembros del cluster DNS (titrit/amazzal).

Ubicación

```
/root/utilidades/directadmin/da_dmarc_bulk.sh
```

Repositorio: `~/utilidades/` (git)

Uso

```
# Ver qué haría sin hacer cambios
./da_dmarc_bulk.sh --dry-run

# Ejecutar en todos los dominios
./da_dmarc_bulk.sh

# Solo un dominio específico
./da_dmarc_bulk.sh --domain ejemplo.com

# Con política quarantine en vez de none
./da_dmarc_bulk.sh --policy quarantine

# Con dirección RUA personalizada (%DOMAIN% se sustituye por el dominio)
./da_dmarc_bulk.sh --rua "postmaster@%DOMAIN%"
```

Opciones

Opción	Descripción	Default
`--dry-run`	Muestra qué haría sin hacer cambios	—
`--policy`	Política DMARC: `none`, `quarantine`, `reject`	`none`
`--rua`	Email para reportes agregados. `%DOMAIN%` = placeholder	`dmarc@%DOMAIN%`
`--domain`	Procesar solo este dominio	todos

Requisitos

- Root en servidor DirectAdmin
- `curl`, `dig`, `python3`
- DirectAdmin binary en `/usr/local/directadmin/directadmin`

Cómo funciona

1. Obtiene credenciales API via `directadmin api-url`
2. Lista usuarios y dominios desde el filesystem (`/usr/local/directadmin/data/users/*/domains/*.conf`)
3. Para cada dominio, consulta `dig +short _dmarc.DOMAIN TXT @127.0.0.1`
4. Si no tiene DMARC, lo crea via `CMD_API_DNS_CONTROL` con impersonación (`admin|usuario`)
5. El cluster DNS sincroniza automáticamente

Detalles técnicos

API DirectAdmin para DNS

- **Endpoint:** `CMD_API_DNS_CONTROL?domain=DOMINIO&json=yes`
- **Impersonación obligatoria:** `admin|usuario:password` — el admin no puede modificar DNS de un dominio directamente, debe impersonar al usuario propietario
- **Formato del value TXT:** las comillas deben ir URL-encoded (`%22`) dentro del valor: `%22v%3DDMARC1...%22`
- **Propagación:** DA incrementa el serial de la zona y los slaves sincronizan via AXFR. Si no sincronizan inmediatamente, ejecutar `rndc notify DOMINIO` en el master

Ficheros que NO son dominios

El script filtra automáticamente ficheros `.hotlink`, `.suspended` y `.letsencrypt` que DA almacena en el mismo directorio que los `.conf` de dominios.

Dominios con DNS externo

Dominios cuyo DNS no está en el servidor DA (ej: Cloudflare) darán error `Domain does not belong to you` o `DNS controlled remotely`. Es esperado — esos DMARC hay que crearlos manualmente en el panel DNS externo.

Ejemplo de salida

```
==========================================
 DirectAdmin Bulk DMARC Creator
==========================================
 Policy:  none
 RUA:     dmarc@%DOMAIN%
==========================================

[OK] elayudante.es (user: elayudante) — DMARC created
[OK] fontaneriafontcal.com (user: ffontcal) — DMARC created
[SKIP] alufasa.com — DMARC already exists
[ERROR] status.castris.com (user: laravel): DNS controlled remotely

==========================================
 Results
==========================================
 Total domains:  91
 Created:        87
 Already exist:  3
 Errors:         1
==========================================
```

Ejecución inicial (2026-03-12)

Servidor	Total	Creados	Ya existían	Errores

kvm456	91	87	4	0
dar	19	15	4	0
srv120	276	275	0	1 (.hotlink)
srv121	139	76	62	1 (.hotlink)
amazzal	2	0	1	1 (DNS externo)
titrit	0	0	0	0

Registro DMARC creado

```
_dmarc  1800  IN  TXT  "v=DMARC1; p=none; rua=mailto:dmarc@DOMINIO"
```

- **`p=none`** — modo monitorización (no rechaza ni pone en cuarentena). Recomendado como primer paso
- **`rua`** — dirección donde llegan los reportes agregados. Requiere que exista el buzón `dmarc@dominio` (o redirección)

Licencia

AGPL-3.0 — GNU Affero General Public License v3.0