# Investigación: Módulos de Apache para Seguridad y Rendimiento

## Propósito

Este documento recoge los módulos de Apache recomendados para reforzar la seguridad y el rendimiento en un entorno compuesto por contenedores Docker (Docker Swarm), balanceo de carga (HAProxy) y máquinas virtuales, con almacenamiento compartido (GlusterFS) y bases de datos MariaDB. Se incluyen comandos de instalación, habilitación, ejemplos de configuración y la justificación técnica para cada módulo.

Nota: Varios de estos comandos requieren privilegios de administrador (sudo). En Docker, integre estas acciones en el Dockerfile o en la imagen base.

## Tabla resumen de módulos recomendados

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Módulo | Propósito | Categoría | Oficial | Justificación para el proyecto |
| mod\_ssl | Cifrado TLS/SSL (HTTPS) | Seguridad | Oficial | Protege comunicaciones; requisito para HTTP/2 seguro. |
| mod\_headers | Gestionar cabeceras HTTP (HSTS, CSP, etc.) | Seguridad | Oficial | Permite aplicar políticas de seguridad via headers. |
| mod\_remoteip | Reemplaza IP cliente cuando hay proxies/HAProxy | Seguridad / Registro | Oficial | Obtiene la IP real del cliente detrás de balanceadores. |
| mod\_security (security2) | WAF - Filtrado y reglas OWASP | Seguridad | No oficial (frecuente) | Filtra solicitudes maliciosas y aplica reglas OWASP. |
| mod\_evasive | Mitigación básica anti-DoS/Rate-limit | Seguridad | No oficial (frecuente) | Bloquea IPs que hacen demasiadas peticiones en corto tiempo. |
| mod\_deflate | Compresión (gzip/deflate) de respuestas | Rendimiento | Oficial | Reduce tamaño de respuestas, mejora latencia y ancho de banda. |
| mod\_expires | Cabeceras de expiración / cache-control | Rendimiento | Oficial | Mejora caché en cliente y reduce cargas repetidas. |
| mod\_cache + mod\_cache\_disk | Cacheo en servidor (almacén en disco) | Rendimiento | Oficial | Sirve respuestas sin regenerarlas; reduce carga de backends. |
| mod\_http2 | Soporte HTTP/2 (multiplexado) | Rendimiento | Oficial | Mejora transporte de múltiples recursos; requiere TLS en navegadores. |
| mod\_status | Página de estado/monitorización (server-status) | Rendimiento/Monitoreo | Oficial | Permite ver uso de procesos, peticiones y diagnosticar cuellos. |
| mod\_proxy + mod\_proxy\_http | Proxy reverso y enrutar a backends (contenerizadores) | Rendimiento/Arquitectura | Oficial | Enruteo a contenedores PHP-FPM, API o servicios internos. |
| mod\_rewrite | Motor de reescritura de URLs y reglas | Seguridad/Utilidad | Oficial | Útil para redirecciones HTTP→HTTPS, bloqueo de rutas y URLs limpias. |

## Detalles por módulo (instalación, habilitación y ejemplo de configuración)

### mod\_ssl

Propósito: Cifrado TLS/SSL (HTTPS)

Categoría: Seguridad

Comandos de instalación / habilitación (Ubuntu/Debian):

sudo a2enmod ssl  
# Generar certificado (ejemplo, LetsEncrypt recomendado):  
# sudo apt install certbot python3-certbot-apache  
# sudo certbot --apache -d ejemplo.tu-dominio  
sudo systemctl restart apache2

Notas: Use certificados de Let's Encrypt en producción; asegure redirecciones HTTP->HTTPS y HSTS si aplica.

* Justificación en el contexto del proyecto: Protege comunicaciones; requisito para HTTP/2 seguro.

### mod\_headers

Propósito: Gestionar cabeceras HTTP (HSTS, CSP, etc.)

Categoría: Seguridad

Comandos de instalación / habilitación (Ubuntu/Debian):

sudo a2enmod headers  
# Ejemplos de seguridad (VirtualHost):  
# Header always set Strict-Transport-Security "max-age=63072000; includeSubDomains; preload"  
# Header set X-Content-Type-Options nosniff  
sudo systemctl restart apache2

* Justificación en el contexto del proyecto: Permite aplicar políticas de seguridad via headers.

### mod\_remoteip

Propósito: Reemplaza IP cliente cuando hay proxies/HAProxy

Categoría: Seguridad / Registro

Comandos de instalación / habilitación (Ubuntu/Debian):

sudo a2enmod remoteip  
# Ejemplo en conf:  
# RemoteIPHeader X-Forwarded-For  
# RemoteIPInternalProxy 10.0.0.0/8  
sudo systemctl restart apache2

* Justificación en el contexto del proyecto: Obtiene la IP real del cliente detrás de balanceadores.

### mod\_security (security2)

Propósito: WAF - Filtrado y reglas OWASP

Categoría: Seguridad

Comandos de instalación / habilitación (Ubuntu/Debian):

sudo apt update  
sudo apt install -y libapache2-mod-security2  
sudo a2enmod security2  
sudo systemctl restart apache2

Notas: descargue y configure OWASP CRS para reglas efectivas (OWASP ModSecurity Core Rule Set).

* Justificación en el contexto del proyecto: Filtra solicitudes maliciosas y aplica reglas OWASP.

### mod\_evasive

Propósito: Mitigación básica anti-DoS/Rate-limit

Categoría: Seguridad

Comandos de instalación / habilitación (Ubuntu/Debian):

sudo apt update  
sudo apt install -y libapache2-mod-evasive  
sudo a2enmod evasive  
sudo systemctl restart apache2

Notas: mod\_evasive requiere ajustar /etc/apache2/mods-available/evasive.conf con parámetros de sensibilidad.

* Justificación en el contexto del proyecto: Bloquea IPs que hacen demasiadas peticiones en corto tiempo.

### mod\_deflate

Propósito: Compresión (gzip/deflate) de respuestas

Categoría: Rendimiento

Comandos de instalación / habilitación (Ubuntu/Debian):

sudo a2enmod deflate  
# Ejemplo (en conf o VirtualHost):  
# AddOutputFilterByType DEFLATE text/html text/plain text/xml text/css application/javascript  
sudo systemctl restart apache2

Notas: exclude browsers antiguos si es necesario; comprimir reduce ancho de banda pero aumenta CPU.

* Justificación en el contexto del proyecto: Reduce tamaño de respuestas, mejora latencia y ancho de banda.

### mod\_expires

Propósito: Cabeceras de expiración / cache-control

Categoría: Rendimiento

Comandos de instalación / habilitación (Ubuntu/Debian):

sudo a2enmod expires  
# Ejemplo:  
# ExpiresActive On  
# ExpiresByType image/jpg "access plus 7 days"  
sudo systemctl restart apache2

* Justificación en el contexto del proyecto: Mejora caché en cliente y reduce cargas repetidas.

### mod\_cache + mod\_cache\_disk

Propósito: Cacheo en servidor (almacén en disco)

Categoría: Rendimiento

Comandos de instalación / habilitación (Ubuntu/Debian):

sudo a2enmod cache cache\_disk  
# Ejemplo básica en VirtualHost:  
# CacheEnable disk /  
# CacheRoot /var/cache/apache2/mod\_cache\_disk  
sudo systemctl restart apache2

Notas: mod\_cache necesita un storage manager (mod\_cache\_disk) y ajuste de htcacheclean para mantenimiento.

* Justificación en el contexto del proyecto: Sirve respuestas sin regenerarlas; reduce carga de backends.

### mod\_http2

Propósito: Soporte HTTP/2 (multiplexado)

Categoría: Rendimiento

Comandos de instalación / habilitación (Ubuntu/Debian):

sudo a2enmod http2  
# En VirtualHost (TLS):  
# Protocols h2 http/1.1  
sudo systemctl restart apache2

Notas: HTTP/2 funciona mejor con TLS (h2) y puede aumentar el uso de memoria por conexión; pruebe y monitorice.

* Justificación en el contexto del proyecto: Mejora transporte de múltiples recursos; requiere TLS en navegadores.

### mod\_status

Propósito: Página de estado/monitorización (server-status)

Categoría: Rendimiento/Monitoreo

Comandos de instalación / habilitación (Ubuntu/Debian):

sudo a2enmod status  
# Config example (apache2.conf or conf-enabled):  
# <Location /server-status>  
# SetHandler server-status  
# Require local # o permitir solo IPs de administración  
# </Location>  
sudo systemctl restart apache2

* Justificación en el contexto del proyecto: Permite ver uso de procesos, peticiones y diagnosticar cuellos.

### mod\_proxy + mod\_proxy\_http

Propósito: Proxy reverso y enrutar a backends (contenerizadores)

Categoría: Rendimiento/Arquitectura

Comandos de instalación / habilitación (Ubuntu/Debian):

sudo a2enmod proxy proxy\_http proxy\_balancer lbmethod\_byrequests  
# Ejemplo (VirtualHost):  
# ProxyPass /api http://127.0.0.1:8080/  
# ProxyPassReverse /api http://127.0.0.1:8080/  
sudo systemctl restart apache2

* Justificación en el contexto del proyecto: Enruteo a contenedores PHP-FPM, API o servicios internos.

### mod\_rewrite

Propósito: Motor de reescritura de URLs y reglas

Categoría: Seguridad/Utilidad

Comandos de instalación / habilitación (Ubuntu/Debian):

sudo a2enmod rewrite  
# Ejemplo para forzar HTTPS en .htaccess o VirtualHost:  
# RewriteEngine On  
# RewriteCond %{HTTPS} !=on  
# RewriteRule ^/?(.\*) https://%{SERVER\_NAME}/$1 [R=301,L]  
sudo systemctl restart apache2

* Justificación en el contexto del proyecto: Útil para redirecciones HTTP→HTTPS, bloqueo de rutas y URLs limpias.

## Referencias y recursos útiles

**Fuentes principales (documentación oficial y guías):**

**Apache mod\_ssl (docs):** https://httpd.apache.org/docs/2.4/mod/mod\_ssl.html

**Apache mod\_http2 (docs):** https://httpd.apache.org/docs/current/mod/mod\_http2.html

**Apache mod\_deflate (docs):** https://httpd.apache.org/docs/current/mod/mod\_deflate.html

**Apache caching modules (mod\_cache/mod\_cache\_disk):** https://httpd.apache.org/docs/current/mod/mod\_cache.html

**Ubuntu - How to install Apache2 (server guide):** https://documentation.ubuntu.com/server/how-to/web-services/install-apache2/

**ModSecurity install guide (Debian/Ubuntu):** https://linuxbabe.com/security/modsecurity-apache-debian-ubuntu

**mod\_evasive installation & config (guide):** https://phoenixnap.com/kb/apache-mod-evasive