apache, bastionado, hardening

# Bastionar Apache

## Instalación

- Mejor compilar que instalar de binarios.
- Mejor en un entorno chroot

## Actualizaciones.

Hay que mantener el servidor actualizado con las últimas actualizaciones. A la hora de actualizar comprobar el changelog para ver si hay incompatibilidades.

- Descargar las actualizaciones de las fuentes oficiales http://httpd.apache.org/download.cgi
- Comprobar el hash y las firmas del fichero que te descargues http://www.apache.org/dist/httpd/KEYS

Si lo hemos instalado desde los repositorios podemos hacer

```
yum update httpd
```

## Desactivar los módulos innecesarios

Para ver los módulos cargados ejecutar

```
 httpd -t -D DUMP_MODULES</code> o bien <sxh bash>grep LoadModule
/etc/httpd/conf/httpd.conf
```

```
grep LoadModule /etc/apache2/apache2.conf
```

para ver los módulos con los que se compiló

```
 httpd -V
```

Todos los módulos que hay para Apache están documentados en la siguiente URL: http://modules.apache.org/

Módulos que se suelen cargar por defecto:

- mod_imap: Que ofrece servicio de mapeo automático de ficheros de índice del lado del servidor.
- mod_include: Habilita los includes de ficheros del lado del seridor. Los .shtml.
- mod_info: Da información sobre el servidor. Los escáneres rastrean la información que ofrece.

Se suele habilitar en pruebas y desarrollo, pero no en producción.
- mod_userdir: Para mapear los directories personales de los usuarios. También está el mod-ldap-userdir para hacerlo vía árboles ldap.
- mod_status: Para tener estadísticas.
- mod_cgi: Ofrece soporte para ejecución de cgis. Si no tienes programas cgi en tu servidor deshabilítalo.
- mod_autoindex: Listados de directorio para cuando no hay archivo por defecto.

Para desactivar un módulo y que no se cargue basta con poner un comentario en la línea que carga el módulo en el fichero de configuración Como minimo eliminar o deshabilitar los siguientes módulos

- mod_negotiation
- mod_user_dir

En CentOS / Redhat (RHEL) / Fedora para deshabilitar un módulo deberemos de renombrar dicho módulo quitándole la extensión **.conf** y reiniciamos el servicio del apache

Por ejemplo para deshabilitar el modulo **mod_python** vamos a la carpeta /etc/httpd/conf.d/ y renombramos el fichero **python.conf** a **python.bak** y reiniciamos el servidor apache

Para habilitar un módulo basta con hacer el proceso contrario. Renombramos el fichero a su extensión **.conf** y reiniciamos el servicio de apache

En Debian / Ubuntu para deshabilitar un módulo podemos usar dos scripts :

- a2enmod es un script para habilitar un módulo dentro de la configuración del apache2.
- a2dismod es un script para deshabilitar un módulo

La forma de deshabilitar un módulo sería ejecutarlo

```
a2dismod {nombre-módulo}
```

y para habilitar un módulo sería con

```
a2enmod {nombre-módulo}
```

con el comando **apachectl configtest** debemos de comprobar que no hemos deshabilitado un módulo que estemos usando

## Instalar módulos

Para agregar **nuevos módulos** mediante el comando **config-status.**

### Ejemplo Instalación de Módulos

Para instalar módulos, el **primer módulo** que debe ser activado es el **módulo para módulos**, esto se realiza mediante el comando:

```
./config.status --activate-module=src/modules/standard/mod_so.c
```

El comando anterior agrega mod_so (el módulo de módulos) a **config.status**; para instalar otros módulos se utilizan parámetros similares:

```
./config.status --enable-module=proxy
```

#### Módulos que pueden aumentar la seguridad

- mod_rewrite
- mod_headers
- mod_setenvif
- mod_security
- mod_auth
- mod_ssl

El **mod_security** hace las veces de firewall de las aplicaciones web y nos permite además monitorizar el tráfico en tiempo real.

El **mod_evasive** nos protege de ataques por fuerza bruta y DDos

Para instalar ambos módulos

```
yum install mod_security mod_evasive
```

<note>reiniciar el servicio httpd para aplicar los cambios </code>

## Permisos

1. Sólo el administrador debe poder modificar los archivos de configuración para que nadie más pueda manipularlos.
2. Crear una cta para arrancar y parar los servicios
3. Crear un grupo para gestionar el servidor

## Suministrar la menor información

Modificamos /etc/httpd/conf/httpd.conf

```
ServerTokens ProductOnly
ServerSignature Off
```

## Quitar el acceso a los listados de directorios

quitar o comentar el acceso a los index

```
## option indexes FlollowSymLinks
```

También podemos podemos editar la configuración y usar la orden options

```
<Directory /var/www/html>
     Options -Indexes
</directory>
```

#### Desactivar la directiva de uso como proxy

```
ProxyRequests off
```

### Directiva FilesMacth Limitar los ficheros a descargar

```
<FilesMatch "\.(old|bak|tgz|sql|inc|tar\.gz|zip|rar)$">
   Order Deny,Allow
   Deny from All
</FilesMatch>
```

### Fortificar con Varnish

http://terminus.ignaciocano.com/k/2011/05/26/mejorando-la-seguridad-de-apache-con-varnish/

### Herramientas

- genera .htaccess para banear visitantes
- otro generador de ficheros .htaccess para banear visitantes
- http://hpantaleev.wordpress.com/2011/09/06/monitorizacion-de-apache-con-apachetop-en-debian-6/

## Referencias


- http://www.rediris.es/cert/doc/reuniones/fs2008/archivo/apache-rediris08.pdf
- http://www.petefreitag.com/item/505.cfm
- http://xianshield.org/guides/apache2.0guide.html
- http://terminus.ignaciocano.com/k/2012/09/21/comprobar-que-no-tenemos-configurado-apache-como-un-proxy-abierto/
- http://httpd.apache.org/docs/2.4/misc/security_tips.html
- http://blog.spiderlabs.com/modsecurity-rules/