

Securitización de apache

Escrito por Administrator

Miércoles, 30 de Marzo de 2016 15:06 - Actualizado Miércoles, 30 de Marzo de 2016 18:42

Apache es el día de hoy uno de los mas populares servidores web, segun las ultimas mediciones alrededor del 50% de las páginas web son servidas por Apache, se encuentra como servidor por defecto en la mayoría de las distribuciones de Linux y es facil de instalar y poner en funcionamiento, por defecto, instala y activa una serie de modulos que no necesariamente utilizaremos y que podrian poner a nuestro sistema en riesgo de ataques.

Aca hay una pequeña guia que nos indica como instalar y configurar varios modulos de Apache, para aumentar la seguridad y disponibilidad de nuestra maquina, por supuesto se deben hacer muchas configuraciones adicionales para aumentar la seguridad y obviamente siempre debemos revisar nuestro código para eliminar potenciales amenazas (Inyeccion SQL, Subida de archivos peligrosos al servidor, etc.).

Las configuraciones propuestas fueron realizadas sobre la ultima versión de Apache de Ubuntu 12.04 LTS (30-03-2016).

Estando como un usuario con privilegios administrativos, realizar los siguientes pasos, para la instalación de los módulos que ayudarán a aumentar la seguridad de Apache 2.x.

Deshabilitar y habilitar módulos de Apache.

Los módulos mínimos para correr apache en mi caso son:

core_module (static)

log_config_module (static)

logio_module (static)

mpm_prefork_module (static)

http_module (static)

so_module (static)

Securitización de apache

Escrito por Administrator

Miércoles, 30 de Marzo de 2016 15:06 - Actualizado Miércoles, 30 de Marzo de 2016 18:42

authz_host_module (shared)

deflate_module (shared)

dir_module (shared)

env_module (shared)

mime_module (shared)

negotiation_module (shared)

reqtimeout_module (shared)

setenvif_module (shared)

rewrite_module (shared) / esté en la mayoría de los casos es necesario para esconder valores en las url.

php5_module (shared) / Este es necesario para poder ejecutar los sitios desarrollados por Inpact.

Estos los instalaremos y configuraremos posteriormente (todos relacionados con seguridad).

evasive20_module (shared)

spamhaus_module (shared)

qos_module (shared)

Para habilitar módulos usaremos el comando `a2enmod <nombre módulo>`, para deshabilitarlos `a2dismod <nombre módulo>`, para ello no se debe incluir "_module", por ejemplo,

```
a2enmod rewrite
```

```
a2dismod autoindex
```

Modulo evasive, permite evitar ataques DDOS

Instalamos con

```
Apt-get -y install libapache2-mod-evasive
```

Creamos la carpeta para los logs y le damos permiso al usuario que ejecuta apache

```
mkdir /var/log/apache/evasive
```

Securitización de apache

Escrito por Administrator

Miércoles, 30 de Marzo de 2016 15:06 - Actualizado Miércoles, 30 de Marzo de 2016 18:42

```
chown -R www-data:www-data /var/log/apache/evasive
```

Editamos el archivo de configuración del módulo, en mi caso particular, agregue la configuración en el mismo archivo que carga el módulo (/etc/apache2/mods-available/mod-evasive.load).

```
DOSHashTableSize 2048
```

```
DOSPageCount 20
```

```
DOSSiteCount 300
```

```
DOSPageInterval 1.0
```

```
DOSSiteInterval 1.0
```

```
DOSBlockingPeriod 10.0
```

```
DOSLogDir "/var/log/apache2/evasive"
```

```
DOSEmailNotify xxxxx@xxxx.xxx
```

Módulo Spamhaus, prevenir ataques de inyección dns

```
sudo apt-get -y install libapache2-mod-spamhaus
```

Creamos el archivo de lista blanca, en el cual agregaremos url y direcciones IP que no queremos que el módulo identifique como atacantes.

```
touch /etc/spamhaus.wl
```

Luego configuramos el módulo editando el archivo de configuración de este (/etc/apache2/mods-available/mod-spamhaus.conf).

```
<IfModule mod_spamhaus.c>
```

```
#MS_Methods
```

```
# Syntax: MS_Methods POST,PUT,OPTIONS
```

```
# Default: POST,PUT,OPTIONS
```

```
#
```

```
# The values admitted are the httpd's methods (GET,POST,etc)
```

Securitización de apache

Escrito por Administrator

Miércoles, 30 de Marzo de 2016 15:06 - Actualizado Miércoles, 30 de Marzo de 2016 18:42

Module verify remote ip address if the method used by the user is present

in the value passed to this variable. Methods must be comma-separated

MS_METHODS POST,PUT,OPTIONS,CONNECT

#MS_WhiteList

Syntax: MS_WhiteList /etc/spamhaus.wl

Default: no value

Path of whitelist file.

After you've edit it, you mustn't reload apache. This file will be read only

when 'data modification time' change. You can add an individual IP address or

subnets with CIDR.

MS_WhiteList /etc/spamhaus.wl

#MS_DNS

Syntax: MS_DNS sbl-xbl.spamhaus.org

Default: sbl-xbl.spamhaus.org

Name server to use for verify is an ip is blacklisted.

Using a local rblndsd instance of sbl-xbl, you can increase query performance

#MS_Dns local.rblndsd.instance.of.sbl-xbl

#MS_CacheSize

Syntax: MS_CacheSize 256

Default: 512

Max value: 8192

This directive can manage the number of cache entries.

MS_CacheSize 512

Securitización de apache

Escrito por Administrator

Miércoles, 30 de Marzo de 2016 15:06 - Actualizado Miércoles, 30 de Marzo de 2016 18:42

```
#MS_CustomError
```

```
# Syntax: MS_CustomError "My custom error message"
```

```
# Default: "Access Denied! Your address is blacklisted. More information about this error may be available in the server error log."
```

```
# A custom error message that allows you to replace default error message with one you create
```

```
MS_CustomError "Access Denied! Your address is blacklisted. More information about this error may be available in the server error log."
```

```
</IfModule>
```

Módulo QOS, calidad de servicio (termina conecciones que no tienen actividad y usan recursos).

```
sudo apt-get -y install libapache2-mod-qos
```

Editamos el archivo de configuración (/etc/apache2/mods-available/mod-spamhaus.conf)

```
<IfModule qos_module>
```

```
# conecciones desde hasta 200 IP distintas
```

```
QS_ClientEntries 200
```

```
# minimum request rate (bytes/sec at request reading):
```

```
#QS_SrvRequestRate          120
```

```
# limits the connections for this virtual host:
```

```
#QS_SrvMaxConn              100
```

```
# allows keep-alive support till the server reaches 600 connections:
```

```
#QS_SrvMaxConnClose         600
```

```
# allows max 50 connections from a single ip address:
```

```
#QS_SrvMaxConnPerIP         50
```

```
#Cantidad maxima de conecciones TCP
```

Securitización de apache

Escrito por Administrator

Miércoles, 30 de Marzo de 2016 15:06 - Actualizado Miércoles, 30 de Marzo de 2016 18:42

MaxClients 256

#deshabilita mantener conexiones abiertas cuando el 70% estan ocupadas

QS_SrvMaxConnClose 180

</IfModule>

Eliminación de información enviada en las cabeceras de respuesta.

Con esto eliminaremos casi toda la información relativa a sistema operativo, versiones y modulos actualmente instalados.

Si hemos instalado el módulo PHP (php5_module), debemos editar el archivo de configuración de php (/etc/php5/apache2/php.ini), y buscamos la siguiente línea,

```
expose_php = On
```

Y la cambiamos por

```
expose_php = Off
```

Ahora eliminaremos la información relacionada con apache, para ello editamos el archivo security de apache (/etc/apache2/conf.d/security), Cambiamos la línea.

```
ServerSignature On
```

por

```
ServerSignature Off
```

Y la línea

```
ServerTokens Full
```

por

```
ServerTokens Prod
```

Con estos cambios al enviar las cabeceras de respuesta solo informaremos al cliente que estamos ejecutando apache.

Adicionalmente activamos IPTABLES.

Creamos un archivo con las reglas en etc y le damos permiso de lectura, ejecución y lectura solo al root.

Securitización de apache

Escrito por Administrator

Miércoles, 30 de Marzo de 2016 15:06 - Actualizado Miércoles, 30 de Marzo de 2016 18:42

touch /etc/iptables

Y agregamos las siguiente líneas.

```
iptables -F # limpiamos reglas actuales de iptables
```

```
iptables -A INPUT -i lo -j ACCEPT # aceptamos todo tipo de peticiones para la interface loop
```

```
iptables -A INPUT -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
```

```
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
```

```
iptables -A INPUT -s 192.168.x.x/24 -p tcp -m tcp --dport 22 -j ACCEPT #permitimos acceso ssh solo desde la LAN
```

```
iptables -A INPUT -s 0.0.0.0/0 -p tcp -m tcp --dport 80 -j ACCEPT #aceptamos peticiones al puerto 80 desde cualquier lugar
```

```
iptables -A INPUT -j DROP # denegamos el acceso a cualquier puerto que no sea 80 ó 22
```