Asegurar Apache

Rafael Ortiz Cáceres
Adrián Álvarez Sáez
Álvaro Pérez Luque

SWAP 14/15

Introducción

En este trabajo se intenta aportar unas directrices y consejos para intentar en la medida de lo posible asegurar un servidor web Apache, concretamente la versión 2.2.22.

Brevemente los contenidos que desarrollamos en este trabajo son:

- Cómo modificar la configuración de Apache y sus medidas de seguridad.
- Autenticación y control de acceso a la información.
- Configuración del protocolo HTTPS, que proporciona cifrado de la comunicación y autenticación del servidor.
- Seguridad básica ante ataque DOS.
- Opciones de configuración de acceso a información importante del servidor.

Primeros pasos - Usuarios

Debemos comprobar que el usuario y grupo que usa el servidor para responder las peticiones no tenga privilegios para acceder a los archivos que no están destinados a ser visible para el mundo exterior.

Estos se definen mediante las directivas **User** y **Group.**

```
adalsa@ubuntu-cliente:~$ egrep "^User|^Group|^SuexecUserGroup" /etc/apache2/apache2.conf /etc/apache
2/sites-available/*
/etc/apache2/apache2.conf:User ${APACHE_RUN_USER}
/etc/apache2/apache2.conf:Group ${APACHE_RUN_GROUP}
```

Utiliza los definidos en las variables de entorno **APACHE_RUN_USER** y **APACHE_RUN_GROUP**, el valor de estas está definido en **/etc/apache2/envvars.**

```
adalsa@ubuntu-cliente:~$ egrep "APACHE_RUN_USER|APACHE_RUN_GROUP" /etc/apache2/envvars
export APACHE_RUN_USER=www-data
export APACHE_RUN_GROUP=www-data
```

Tanto el usuario **www-data** como el grupo **www-data** son creados automáticamente en el proceso de instalación.

Permisos del directorio raíz

Debemos asegurar los archivos contenidos en la carpeta raíz del servidor para que estos solo puedan ser modificados por el root.

Para ello ejecutamos los siguientes órdenes:

```
# chown --recursive 0:0 /etc/apache2
```

```
# chmod --recursive 750 /etc/apache2
```

Con la primera orden establecemos como usuario y grupo propietario a root, la segunda orden concede todos los privilegios al usuario root, privilegios de lectura y ejecución para el grupo root, y ningún permiso para el resto de usuarios.

Deshabilitar módulos innecesarios

Para ver los módulos cargados en apache podemos usar:

```
# apache2ctl -M
```

Tambíen podemos verlos listando el contenido del directorio ***mods-enabled***, en este directorio se encuentran enlaces simbólicos a los módulos disponibles en el directorio ***mods-available***.

```
root@ubuntu-cliente:/etc/apache2# ls -Al /etc/apache2/mods-enabled/
total 0
lrwxrwxrwx 1 root root 28 May 24 17:59 alias.conf -> ../mods-available/alias.conf
lrwxrwxrwx 1 root root 28 May 24 17:59 alias.load -> ../mods-available/alias.load
lrwxrwxrwx 1 root root 33 May 24 17:59 auth_basic.load -> ../mods-available/auth_basic.load
lrwxrwxrwx 1 root root 33 May 24 17:59 authn_file.load -> ../mods-available/authn_file.load
lrwxrwxrwx 1 root root 36 May 24 17:59 authz_default.load -> ../mods-available/authz_default.load
```

Deshabilitar módulos innecesarios

Como medida de seguridad es aconsejable deshabilitar los módulos que no se vayan utilizar, así evitamos posibles ataques a vulnerabilidades de módulos que no son necesarios y además ahorraremos recursos.

Para habilitar o deshabilitar un módulo disponemos de los siguientes comandos:

```
# a2dismod proxy_ftp
```
//Deshabilita el módulo proxy_ftp

```
# a2enmod proxy_ftp
```
//Habilita el módulo proxy_ftp

Para activar la nueva configuración es necesario reiniciar Apache:

```
# service apache2 restart
```

Archivos de configuración

- **/etc/apache2/apache2.conf:** Es el archivo principal contiene la mayoría de directivas de configuración de apache, se aconseja no modificarlo directamente, e incluir en él los archivos de configuración, para evitar problemas al realizar actualizaciones de apache y tener un mayor control y organización.

- **/etc/apache2/mods-enabled:** en este directorio se encuentran los archivos que gestionan la carga y configuración de los distintos y posibles módulos de apache.

- **/etc/httpd/httpd.conf:** este archivo contiene información y directivas aplicables que controlan las funciones del servidor.

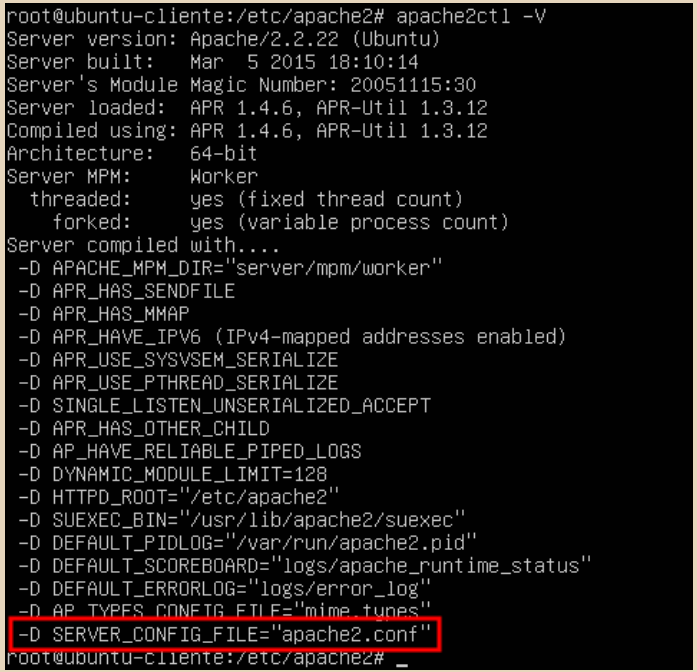
- **/etc/apache2/ports.conf:** contiene las directivas y reglas de control sobre los puertos que usará apache.

- **/etc/apache2/sites-enabled:** este directorio contiene los archivos de configuración de cada virtual host.

Configuración del servidor

Podemos consultar la ubicación del fichero de configuración principal mediante la siguiente orden:

```
apahce2ctl -V
```



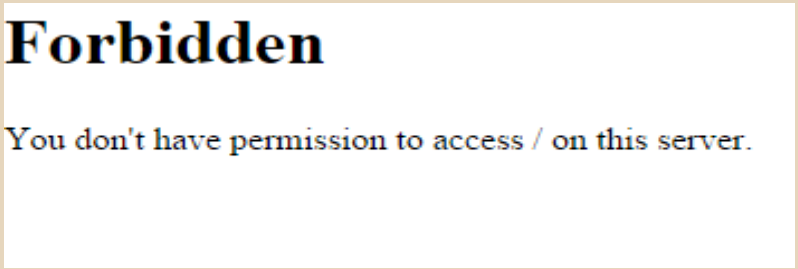
Observando la salida, podemos ver la ruta del archivo de configuración principal:

- SERVER_CONFIG_FILE=”apache2.conf”

Ocultamiento de información

Una de las medidas que se pueden tomar es ocultar toda información de la versión de Apache que se muestra en el pie de página de los documentos generados por el servidor. Para ello utilizamos las siguientes directivas:

`ServerSignature Off`



`ServerTokens Prod`

```
root@ubuntu-cliente:/etc/apache2# curl -i http://127.0.0.1/pagina_inexistente
HTTP/1.1 404 Not Found
Date: Sun, 31 May 2015 01:46:00 GMT
Server: Apache
Vary: Accept-Encoding
Content-Length: 216
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
<p>The requested URL /pagina_inexistente was not found on this server.</p>
</body></html>
root@ubuntu-cliente:/etc/apache2# _
```

Deshabilitar el listado de ficheros

Si solicitamos a Apache un directorio en vez de un archivo, éste nos devolverá un listado de todos los archivos del directorio.

Podemos controlar este comportamiento y otras características para un directorio en particular mediante la directiva *Options*.

En nuestro caso hemos usado el más restrictivo que es *none* para el directorio raíz de este modo todas las características incluidas el índice de carpetas estarán desactivadas.

```
 GNU nano 2.2.6                File: sites-enabled/000-default

<VirtualHost *:80>
        ServerAdmin webmaster@localhost

        DocumentRoot /var/www
        <Directory />
                Order deny,allow
                Deny from all
                Options None
                AllowOverride None
        </Directory>
```

Limitar el acceso a archivos por extensión

Nos puede interesar bloquear el acceso a determinados ficheros que necesariamente deben estar dentro del ***DocumentRoot***, pero no están destinados a ser accedidos por el público.

Para denegar el acceso a cierto tipo ficheros, por ejemplo utilizaremos la siguiente directiva que deniegue el acceso a todos los ficheros que comiencen por ***.ht ,*** restringiendo en este caso acceso a archivos como ***htacces y htpasswd***.

```
<Files ~ "^\.ht">
      Order allow,deny
      Deny from all
</Files>
```

Para denegar el acceso a un directorio se utiliza la directiva ***DirectoryMatch***:

```
<DirectoryMatch /Backup/>
      Order Allow,Deny
      Deny from all
</DirectoryMatch>
```

Control de acceso a archivos publicados

Es recomendable denegar el acceso a todos los directorios por defecto y permitir el acceso sólo al directorio especificado en el DocumentRoot explícitamente.

La directiva ***Directory*** sirve para aplicar una serie de directivas sobre la carpeta especificada y a todas sus subcarpetas.
Las directivas ***Allow*** y ***Deny*** son utilizadas para permitir o denegar respectivamente el acceso a un directorio y ***Order*** especifica el orden en el que serán evaluadas.

```
 GNU nano 2.2.6                File: sites-enabled/000-default

<VirtualHost *:80>
        ServerAdmin webmaster@localhost

        DocumentRoot /var/www
        <Directory />
                Order deny,allow
                Deny from all
                Options None
                AllowOverride None
        </Directory>
        <Directory /var/www/>
                AllowOverride None
                Order allow,deny
                Allow from all
        </Directory>
```

Evitar la resolución de enlaces simbólicos

Para este propósito usaremos la directiva ***Options***, que permite los parámetros:

- **ExecCGI:** se permite la ejecución de scripts CGI.
- **FollowSymLinks:** el servidor seguirá los enlaces simbólicos.
- **Includes:** se permiten incluir Server-side.
- **IncludesNOEXEC:** se permiten incluir Server-side pero se deshabilitan las órdenes #exec y #exec CGI.
- **Indexes:** Si una URL solicita un directorio y no existe DirectoryIndex
- **MultiViews:** se permiten mostrar contenido negociado en función de diversos valores.
- **SymLinksIfOwnerMatch:** Se sigue un enlace simbólico sólo si los propietarios del enlace y del destino coinciden.
- **All:** todas las opciones salvo MultiViews.
- **None:** No permite ninguna opción.

Apache puede recibir una petición a un archivo que es, a nivel de sistema operativo del servidor, un enlace simbólico y devolver el archivo apuntado por él aunque se encuentre fuera del DocumentRoot.
Para deshabilitar esta funcionalidad se utiliza la directiva:

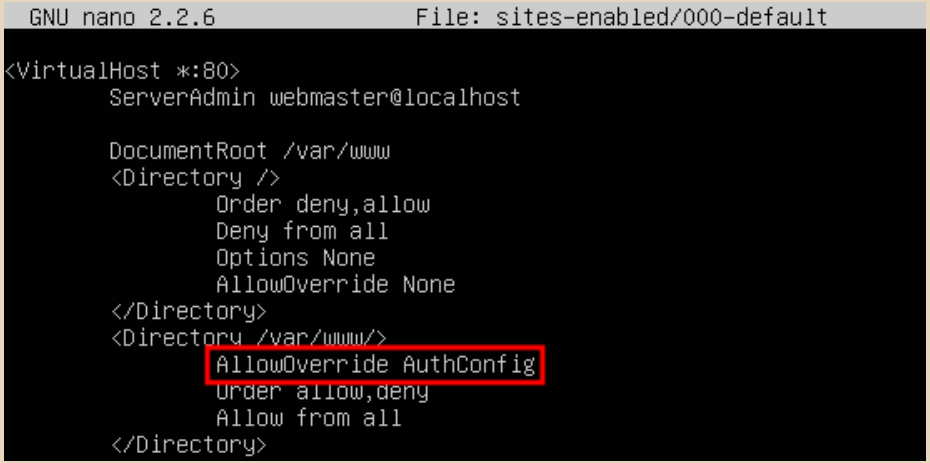
```
Options -FollowSymlinks +SymLinksIfOwnerMatch
```

Autenticación, autorización y control de acceso

Podemos activar un proceso de autenticación para proteger un recurso, solicitando las credenciales al intentar acceder a él.

Para este propósito usaremos archivos ***htaccess***, este archivos permite definir directivas de configuración para un directorio en concreto, la carpeta a la que afectarán estas directivas será en el que está ubicado el archivo ***.htaccess***.

Para que estos archivos puedan sobreescribir las directivas del fichero de configuración principal que atañen a la seguridad usaremos la siguiente directiva:



Autenticación, autorización y control de acceso

Primero crear el fichero que contendrá las contraseñas añadiendo un usuario, para ello usamos el comando ***htpasswd***.

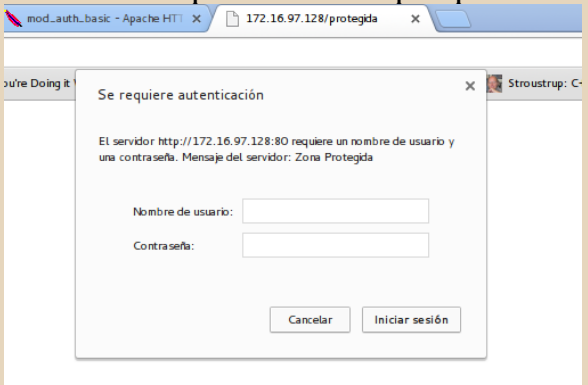
```
root@ubuntu-cliente:/etc/apache2# htpasswd -c /etc/apache2/passwd adrian
New password:
Re-type new password:
Adding password for user adrian
```

Una vez creado el fichero de contraseñas crearemos un fichero ***.htaccess*** en la directorio que deseamos proteger.

Con estas directivas hemos especificado un método de autenticación basado en credenciales en texto plano, cuya ruta es ***/etc/apache2/passwd*** y con la directiva ***Require valid-user*** especificamos que puede acceder cualquier usuario que aparezca en el archivo de contraseñas.

```
 GNU nano 2.2.6              File: /var/www/protegida/.htaccess

AuthType Basic
AuthName "Zona Protegida"
AuthUserFile "/etc/apache2/passwd"
Require valid-user
```



Control de acceso

El módulo ***mod_auth_host***, permite controlar el acceso a cualquier recurso basándose en la dirección del host del visitante.

```
a2enmod authz_host
```

Para controlar el acceso utilizaremos la directiva ***Require*** que admite dos usos:

```
Require host address
Require ip ip.address
```

Por ejemplo para negar el acceso a una ip que ha estado atacando:

```
<RequireAll>
     Require all granted
     Require not ip 217.216.37.42
</RequireAll>
```

Incluso podemos bloquear el acceso desde un dominio:

```
Require not host ugr.es
```

Seguridad ante ataques DOS

Un ataque de denegación de servicio, básicamente consiste en saturar los recursos del sistema impidiendo a consecuencia el acceso a los usuarios legítimos, algunas de las directivas que nos ayudan a prevenir este tipo de ataques se muestran a continuación.

-**Directiva Timeout:** Esta directiva define el tiempo que espera apache para ciertas circunstancias I/O. Este valor viene por defecto establecido en 300 seg, pero en caso de que nuestro servidor sea susceptible de ataques DOS podemos reducir a unos pocos segundos, por ejemplo:

```
Timeout 10
```

-**Directiva KeepAliveTimeOut:** La directiva KeepAliveTimeout establece el número de segundos que el servidor esperará tras haber dado servicio a una petición, antes de cerrar la conexión. Se aconseja establecer un valor bajo, en nuestro caso por ejemplo 5 segundos.

```
KeepAliveTimeout 5
```

Seguridad ante ataques DOS

-Directiva MaxRequestWorkers: Modificando el número máximo de peticiones que puede manejar simultáneamente el servidor podemos evitar que el servidor se quede sin recursos debido a un alto número de peticiones.

```
MaxRequestWorkers 256
```

-RequestReadTimeOut: Esta directiva permite limitar el tiempo que un cliente puede tardar para enviar la petición, permite modificar individualmente el tiempo de espera para la cabecera y el cuerpo.

```
RequestReadTimeout header=10 body=20
```

-Directiva LimitRequestBody: Con esta directiva podemos especificar el número de bytes que se permite en el cuerpo de una petición.

```
LimitRequestBody 102400
```

SSL en Apache

En primer lugar activaremos el módulo ssl ejecutando el siguiente comando:

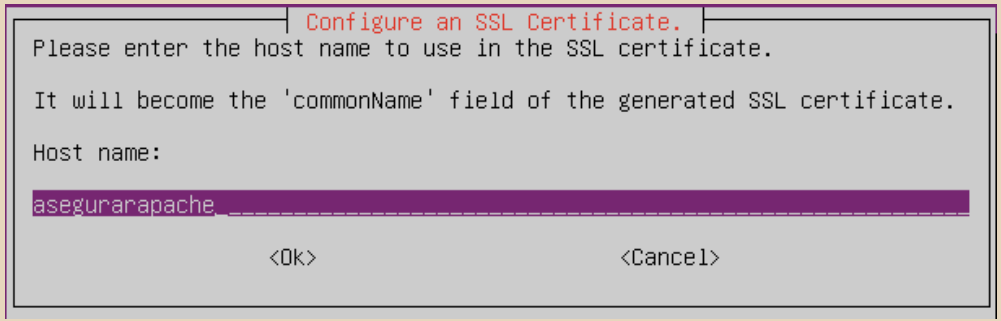
```
#a2enmod ssl
```

También debemos activar el sitio ***default--ssl*** que viene incluido por defecto en Apache, para ello usamos el comando:

```
#a2ensite default_ssl
```

Generar un certificado firmado por nosotros mismos, para ello necesitamos el paquete ***ssl-cert***.

```
make-ssl-cert /usr/share/ssl-cert/ssleay.cnf /etc/apache2/ssl/certificado.crt
```



Hay que comprobar que el certificado solo tenga permisos de lectura para root.

```
root@ubuntu-cliente:/etc/apache2# ll /etc/apache2/ssl/certificado.crt
-rw------- 1 root root 2693 May 31 20:04 /etc/apache2/ssl/certificado.crt
```

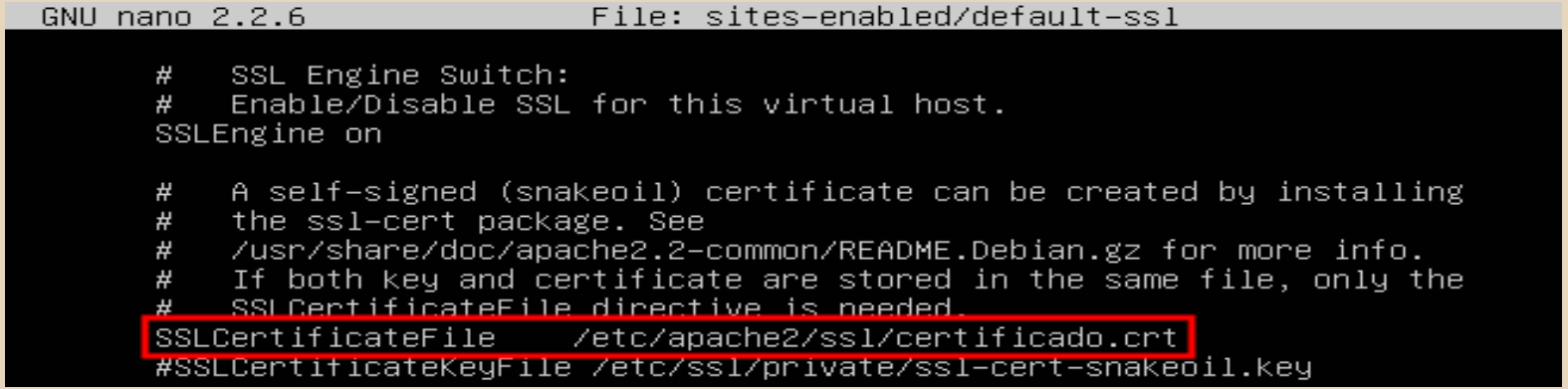
SSL en Apache

Por último tendremos que editar el fichero de configuración del virtual host ssl.

Añadimos la directiva ***SSLCertificateFile*** apuntando al certificado ***.crt*** que acabamos de crear, no es necesario usar la directiva ***SSLCertificateKeyFIle*** ya que la key esta ya incluida en el archivo ***.crt***.

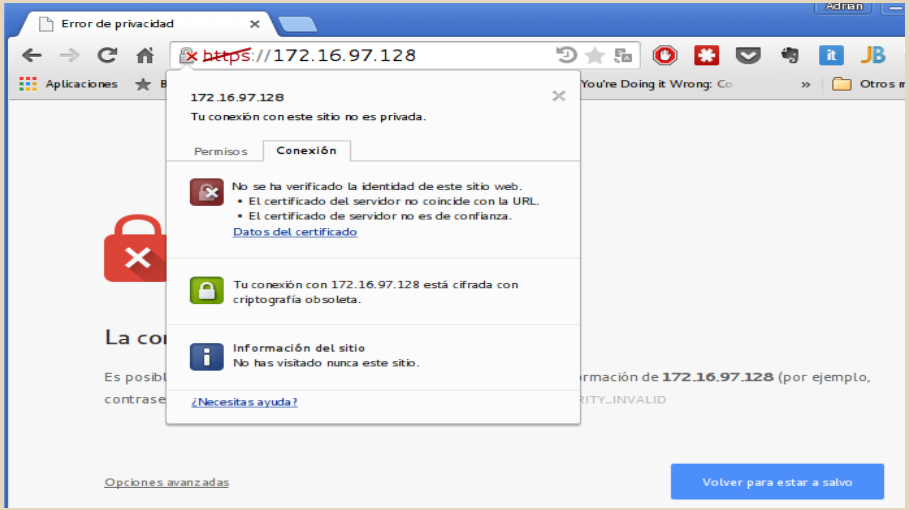
-Reiniciamos el servidor para que los cambios tengan efecto.

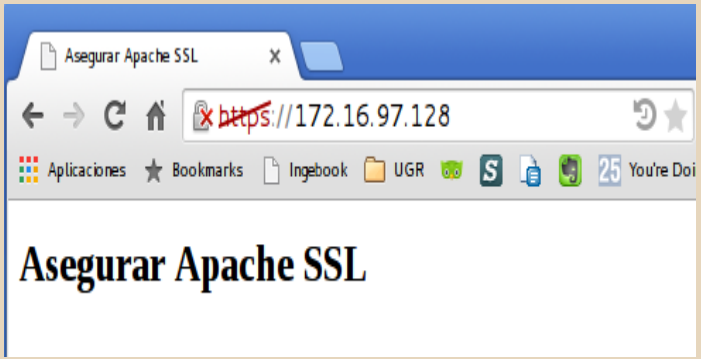
#apache2ctl restart



SSL en Apache

Para probar que todo se ha configurado correctamente probamos a acceder desde un navegador especificando el protocolo ***https://***.
La advertencia que se ve en la imagen se debe a que el certificado que usamos está firmado por nosotros mismo y no por una autoridad reconocida por el explorado. Una vez que le indiquemos al explorador que queremos continuar se mostrará correctamente la página principal que se encuentre en el ***DocumentRoot*** del virtual host ssl.





Bibliografía

- Apache 2.2 Security Tips (http://httpd.apache.org/docs/2.2/misc/security_tips.html)
- Apache 2.2 Documentation (http://httpd.apache.org/docs/2.2/en/)
- Apache Web Server Hardening Security (http://geekflare.com/apache-web-server-hardening-security/)
- En el repositorio de github, se puede encontrar estas misma presentación además de un documento pdf, donde se describe todo el proceso de instalación y configuración de forma más detallada.