Seguridad en servidores HTTP Apache.
Generación de certificado SSL.

Manuel Callejón Roldán
Sergio Rodríguez Marín

Asignatura Servidores Web de Altas Prestaciones
Grado en Ingeniería Informática
Universidad de Granada

15 – Mayo - 2016

ÍNDICE

1. ¿Qué es HTTP Apache?

El **servidor HTTP Apache** es un servidor web HTTP de código abierto, para plataformas Unix, Microsoft, Macintosh y otras, más utilizado y líder con el mayor número de instalaciones a nivel mundial por delante de otras soluciones como el IIS (Internet Informationn Server) de Microsoft (Según sondeos, al rededor del 70% de los sitios web en internet están manejados por Apache).

Es muy robusto y destaca por su seguridad y rendimiento. Además es desarrollado y mantenido por una comunidad de usuarios bajo la supervisión de la Apache Software Foundation dentro del proyecto HTTP Server.

1.1 ¿Qué es un servidor Web y qué hace exactamente?

De forma muy breve, la misión de un servidor Web es crítica, se encarga de aceptar las peticiones y recursos de páginas web y gestionar su entrega o denegación. Esto implica muchas funcionalidades que se deben cubrir, como por ejemplo:

- Atender de manera eficiente un gran número de peticiones HTTP, dándose el caso de peticiones simultáneas.
- Seguridad en el acceso a ficheros que se quieran 'exponer', autentificaciones de usuarios y filtrado de peticiones según el origen.
- Manejo de errores que se hayan producido, manteniendo informado al visitante.

1.2 ¿Qué nos permite hacer Apache?

Además de todas las funcionalidades anteriores, Apache nos permite configurar un Hosting Virtual (Alojamiento web) basado en Ips o en nombres, es decir, tener varios sitios web en un mismo equipo o establecer distintos niveles de control de acceso a la información incluyendo el soporte a cifrado SSL utilizando protocolo seguro **HTTPS.**

Apache nos permite cambiar la funcionalidad y características de los servicios ofrecidos gracias a que es de *código abierto*. Todas estas funcionalidades podemos encontrarlas en:

http://httpd.apache.org/docs/2.0/es/mod/directives.html

Apache cuenta con archivos Log, que poseen registros de información global del sistema, como errores y eventos producidos en un determinado momento.

Otra de las ventajas de Apache es que soporta una gran variedad de lenguajes de programación que nos facilitaran el trabajo a la hora de crear y desarrollar páginas webs dinámicas.

2. Problemas de seguridad en Apache

Como hemos comentado antes, Apache en el servidor Web más utilizado en la actualidad, pero al igual que su competencia por parte de Microsoft, también tiene ciertas vulnerabilidades (aunque bastantes menos).

Respecto al tiempo de vida de Apache, pocos fallos de seguridad ha tenido pero vamos a analizar algunos de ellos:

- El grupo X-Force descubrió una vulnerabilidad en las versiones actuales del servidor. En la que era posible tomar el control de un servidor Web basado en Apache, permitiendo modificar contenidos web e incluso ataques DoS (Denegación de Servicio).
- En la versión 1.3 de Apache el problema causa un desbordamiento en la pila, causando una violación de segmento, haciéndolo vulnerable a dichos ataques.
- En la versión 2.0 se solventó este problema, impidiendo al atacante ejecutar código de su elección en el servidor. En cambio, si se ha configurado manualmente el modelo de ejecución multitarea usando hilos de control en el archivo de configuración de Apache el servidor sí se verá afectado. Es recomendable comprobar esta opción.

Los sistemas vulnerables son todas las versiones anteriores a Apache 1.3.25 y a Apache 2.0.38.

Todas estas debilidades de Apache están siendo solventadas por el Apache Group actualizando a una versión igual o superior a la 1.3.26 y 2.0.39.

3. Configuración de seguridad en Apache

3.1. Directivas de configuración.

El servidor Web Apache posee una serie de archivos que permiten configurar al servidor con la funcionalidad deseada. Esta configuración se realiza a través de directivas o reglas en archivos de configuración concretos. Dependiendo de la distribución de linux varian el número, las rutas y los nombres de los archivos de configuración.

En una distribución Linux, usando el comando *apache2ctl -V* podemos ver los datos sobre la instalación como por ejemplo el archivo de configuración.

```
Server version: Apache/2.2.22 (Ubuntu)
Server built:   Jul 22 2014 14:37:02
Server's Module Magic Number: 20051115:30
Server loaded:  APR 1.4.6, APR-Util 1.3.12
Compiled using: APR 1.4.6, APR-Util 1.3.12
Architecture:   32-bit
Server MPM:     Prefork
  threaded:     no
    forked:     yes (variable process count)
Server compiled with....
 -D APACHE_MPM_DIR="server/mpm/prefork"
 -D APR_HAS_SENDFILE
 -D APR_HAS_MMAP
 -D APR_HAVE_IPV6 (IPv4-mapped addresses enabled)
 -D APR_USE_SYSVSEM_SERIALIZE
 -D APR_USE_PTHREAD_SERIALIZE
 -D SINGLE_LISTEN_UNSERIALIZED_ACCEPT
 -D APR_HAS_OTHER_CHILD
 -D AP_HAVE_RELIABLE_PIPED_LOGS
 -D DYNAMIC_MODULE_LIMIT=128
 -D HTTPD_ROOT="/etc/apache2"
 -D SUEXEC_BIN="/usr/lib/apache2/suexec"
 -D DEFAULT_PIDLOG="/var/run/apache2.pid"
 -D DEFAULT_SCOREBOARD="logs/apache_runtime_status"
 -D DEFAULT_LOCKFILE="/var/run/apache2/accept.lock"
 -D DEFAULT_ERRORLOG="logs/error_log"
 -D AP_TYPES_CONFIG_FILE="mime.types"
 -D SERVER_CONFIG_FILE="apache2.conf"
```

3.2. Tipos de directivas de configuración.

3.2.1 . Directivas ServerSignature, ServerTokens y HTTP Trace

A través de estas directivas podemos evitar mostrar información sobre la versión de Apache ya que si el atacante sabe en qué versión hay una brecha de seguridad puede intentar ese determinado ataque para esa versión en concreto. Para ocultar esta información entramos en el archivo */etc/apache2/conf.d/security* y establecemos los valores:

- ServerSignature Off

```
#
# Optionally add a line containing the server version and virtual host
# name to server-generated pages (internal error documents, FTP directory
# listings, mod_status and mod_info output etc., but not CGI generated
# documents or custom error documents).
# Set to "EMail" to also include a mailto: link to the ServerAdmin.
# Set to one of:  On | Off | EMail
#
#ServerSignature Off
ServerSignature Off
```

- ServerTokens ProductOnly

```
#
# ServerTokens
# This directive configures what you return as the Server HTTP response
# Header. The default is 'Full' which sends information about the OS-Type
# and compiled in modules.
# Set to one of:  Full | OS | Minimal | Minor | Major | Prod
# where Full conveys the most information, and Prod the least.
#
#ServerTokens Minimal
#ServerTokens OS
#ServerTokens Full
ServerTokens Prod
```

La directiva **ServerTokens** configura lo que te devuelve la cabecera de la petición HTTP, así como da información sobre los modulos compilados. Un atacante podría utilizar esta información, por ello se recomienda esta modificación en el archivo.

La directiva **ServerSignature** configura la versión del servidor y la del host virtual. También es recomendable su inhabilitación para evitar ataques con la información devuelta.

Otra directiva es el HTTP Trace, utilizada para devolver toda la información recibida. Un atacante puede modificarlo para que devuelvas las cookies pudiendo así suplantar la sesión. Es recomendable deshabilitarla por cuestiones de seguridad modificando la línea del archivo:

- TraceEnable Off

```
#
# Allow TRACE method
#
# Set to "extended" to also reflect the request body (only for testing and
# diagnostic purposes).
#
# Set to one of:  On | Off | extended
#
TraceEnable Off
#TraceEnable On
```

3.2.2. Configuración global de Apache

Una opción de configuración crucial de Apache es que corra bajo el usuario y grupo Apache, ya que uno de los errores más comunes es ejecutarlo como *root*. Para modificar esto entramos en el archivo */etc/apache2/apache2.conf* y buscamos lo indicado en la siguiente foto:

```
# These need to be set in /etc/apache2/envvars
User ${APACHE_RUN_USER}
Group ${APACHE_RUN_GROUP}
```

y sustituirlo por:

```
# These need to be set in /etc/apache2/envvars
User apache
Group apache
```

3.2.3. Protección contra ataques DoS.

Un ataque DoS (Denegación de Servicio) es un tipo de ataque a un servidor que causa que un servicio sea inaccesible a los usuarios, es decir, provoca la pérdida de la conectividad de la red. Para producir un ataque DoS basta con lanzar multitud de peticiones al servidor.

Para evitar estos ataques, Apache cuenta con la directiva *limitRequestBody*, que restringe la cantidad de datos (bytes) que se permite que tenga el cuerpo de una petición.

Dentro del archivo *httpd.conf* establecemos la directiva a un tamaño que no colapse el servidor.

- LimitRequestBody 1024 #Establecemos el tope a 1KB

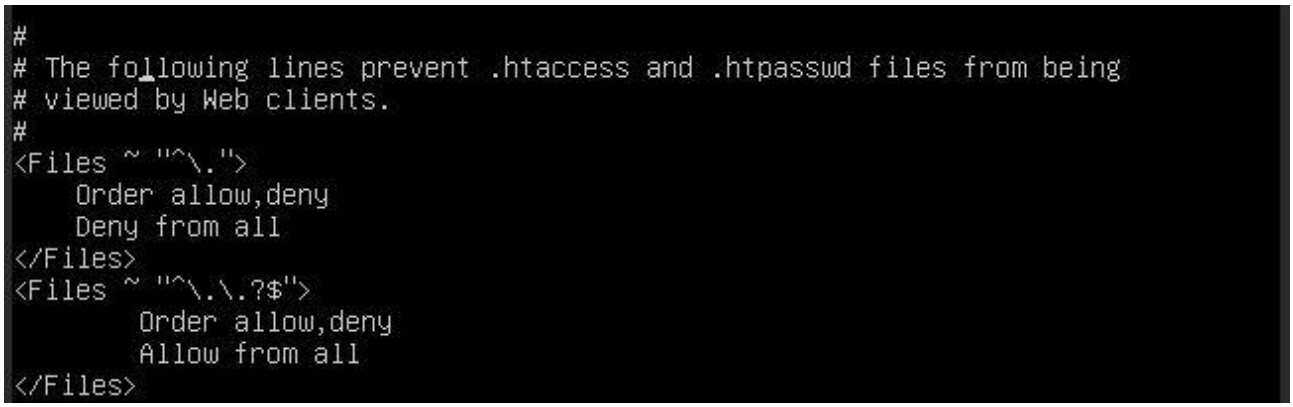
3.2.4. Denegar accesos a directorios o archivos ocultos

En el archivo */etc/apache2/conf.d/security* podemos restringir el acceso a los directorios de nuestro servidor.

Por ejemplo, si en el directorio */images* tenemos diversas imágenes, con la dirección url *http://<IP>/images/* podemos acceder a todas las imágenes del directorio. Para evitar esta situación modificamos la directiva *directory*.

Para evitar acceder a archivos ocultos, como por ejemplo .htaccess que muestra la configuración de los directorios y permite realizar cambios en la configuración. Usamos la directiva *Files* que nos permite no mostrar los archivos.

Para ello en el archivo *apache2.conf* modificamos la directiva *files* de la siguiente manera:



4. Crear certificado SSL

Un certificado SSL sirve para brindar seguridad al visitante en una web, una manera de decirle a los usuarios que el sitio web es auténtico, real y fiable para ingresar sus datos personales. Las siglas SSL corresponden a Secure Socket Layer (Capa de conexión segura), que es un protocolo de seguridad en el que la transmisión de los datos entre cliente y servidor, y viceversa, es totalmente cifrada o encriptada.

Para crear el certificado SSL vamos a usar OpenSSL que tiene el mismo nivel de cifrado que cualquier proveedor.

Para crearlo vamos a usar el sistema operativo Ubuntu 14.04.

4.1. Instalación de OpenSSL

Empezamos instalando openSSL con el comando *sudo apt-get install openssl.*

```
sergio@N53SV:~$ sudo apt-get install openssl
[sudo] password for sergio:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  account-plugin-windows-live cli-common conky conky-std libaudclient2
  libbonoboui2-0 libbonoboui2-common libgdiplus libglade2-0 libgnomecanvas2-0
  libgnomecanvas2-common libgnomeui-0 libgnomeui-common libid3tag0 libimlib2
  libmono-2.0-1 libmono-2.0-dev libmono-profiler libmonoboehm-2.0-1
  libmonoboehm-2.0-dev libmonosgen-2.0-1 libupstart1 libxmmsclient6 mono-jay
  monodoc-manual realpath
Use 'apt-get autoremove' to remove them.
Se actualizarán los siguientes paquetes:
  openssl
1 actualizados, 0 se instalarán, 0 para eliminar y 166 no actualizados.
Necesito descargar 490 kB de archivos.
Se utilizarán 0 B de espacio de disco adicional después de esta operación.
Des:1 http://es.archive.ubuntu.com/ubuntu/ trusty-updates/main openssl amd64 1.0.1f-1ubuntu2.19 [490 kB]
Descargados 490 kB en 7seg. (63,2 kB/s)
(Leyendo la base de datos ... 255342 ficheros o directorios instalados actualmente.)
Preparing to unpack .../openssl_1.0.1f-1ubuntu2.19_amd64.deb ...
Unpacking openssl (1.0.1f-1ubuntu2.19) over (1.0.1f-1ubuntu2.18) ...
Processing triggers for man-db (2.6.7.1-1ubuntu1) ...
Configurando openssl (1.0.1f-1ubuntu2.19) ...
```

4.2. Crear una llave privada

Esta llave privada servirá para la generación del certificado. Este dependerá de la llave para la implementación del mismo en cualquier servicio que necesite una conexión segura. Vamos a generar una llave de 2048 bits. Usamos el comando *openssl genrsa -out server.key 2048.*

La opción *genrsa* nos sirve para generar una clave privada RSA.

El parámetro *-out server.key* genera el archivo server.key con la llave.

2048 indica la cantidad de bits que va a usar la llave.

```
sergio@N53SV:~$ openssl genrsa -out server.key 2048
Generating RSA private key, 2048 bit long modulus
...................+++
.............................................+++
e is 65537 (0x10001)
```

4.3. Crear una solicitud de firma de certificado (CSR)

En el CSR se definen datos como el dominio, organización, información de contacto, entre otros. Estos datos deben ser correctos ya que nos van a identificar.

Para generar el CSR hay que ejecutar el comando *openssl req -new -key server.key -out server.csr*

El parámetro *req* indica que vas a generar el CSR.

-new va a generar un nuevo CSR.

-key server.key indica que va a usasr la llave generada anteriormente

-out server.csr indica que genera un archivo CSR

Al ejecutar el comando aparacerán una serie de preguntas sobre la empresa:

```
sergio@N53SV:~$ openssl req -new -key server.key -out server-local.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ES
State or Province Name (full name) [Some-State]:Granada
Locality Name (eg, city) []:Granada
Organization Name (eg, company) [Internet Widgits Pty Ltd]:SWAP-1516
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:172.20.73.11
Email Address []:sergio93@correo.ugr.es

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

4.4. Generando el certificado SSL

Para crear el certificado SSL vamos a usar la llave y el CSR generado anteriormente. Usaremos el comando *openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt*

```
sergio@N53SV:~$ openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
Signature ok
subject=/C=SP/ST=Granada/L=Granada/O=Practica-SWAP1516/CN=www.pavocejudo.hol.es/emailAddress=sergiobasket93@gmail.com
Getting Private key
```

El parámetro *-days 365* indica cuando expira el certificado.

4.5. Configurar el certificado SSL en Apache

En primer lugar copiaremos los archivos a la carpeta */etc/ssl/certs* con los comandos que se ven a continuación y habilitamos el módulo SSL en Apache

```
sergio@N53SV:~$ sudo cp server.crt /etc/ssl/certs/ssl.crt
sergio@N53SV:~$ sudo cp server.key /etc/ssl/certs/ssl.key
sergio@N53SV:~$ sudo a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create self-signed certificates.
To activate the new configuration, you need to run:
  service apache2 restart
sergio@N53SV:~$
```

Una vez habilitado el módulo, editamos el archivo *vhosts* con *sudo nano /etc/apache2/sites-available/default-ssl.* Si hay algún contenido en dicho archivo, se reemplaza por lo siguiente:

```
<IfModule mod_ssl.c>
  <VirtualHost _default_:443>
    ServerName 172.20.73.11
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
```

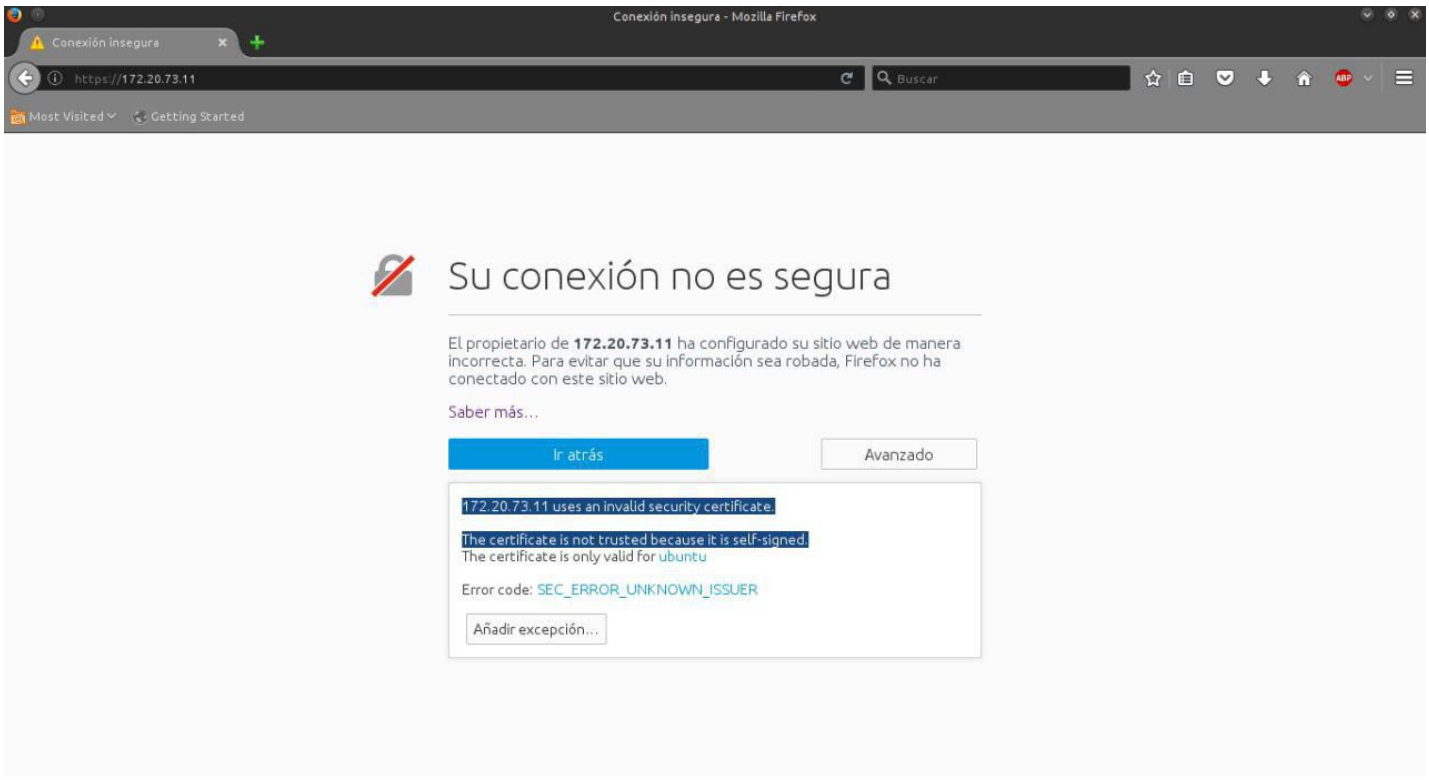
```
    <Directory /var/www/>
      Options -Indexes FollowSymLinks MultiViews
      AllowOverride None
      Order allow,deny
      allow from all
    </Directory>
    ErrorLog ${APACHE_LOG_DIR}/error.log
    LogLevel warn
    CustomLog ${APACHE_LOG_DIR}/ssl_access.log combined
    SSLEngine on
    SSLCertificateKeyFile /etc/ssl/certs/ssl.key
    SSLCertificateFile /etc/ssl/certs/ssl.crt
    #SSLCACertificateFile /etc/ssl/certs/bundle.crt
    BrowserMatch "MSIE [2-6]" \
      nokeepalive ssl-unclean-shutdown \
      downgrade-1.0 force-response-1.0
    # MSIE 7 and newer should be able to use keepalive
    BrowserMatch "MSIE [17-9]" ssl-unclean-shutdown
  </VirtualHost>
</IfModule>
```

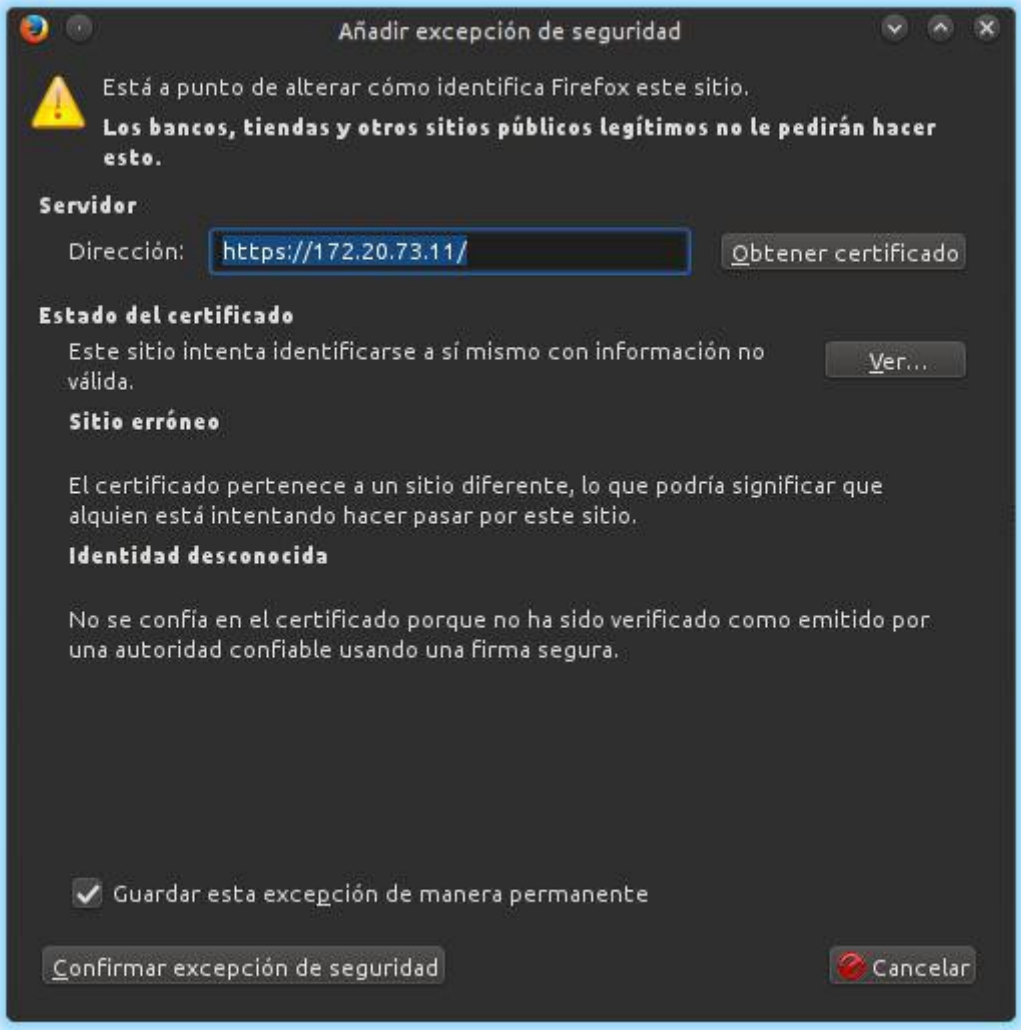
En el apartado S*erverName* se escribe la dirección web.

Una vez configurado el archivo vamos a habilitar y a reiniciar el servicio de Apache con los comandos:

- *sudo a2ensite default-ssl*
- *sudo service apache2 reload*

Ya tenemos el certificado SSL configurado y funcionando.


Conexión insegura - Mozilla Firefox
Conexión insegura
https://172.20.73.11
Buscar
Most Visited
Getting Started
Su conexión no es segura
El propietario de 172.20.73.11 ha configurado su sitio web de manera incorrecta. Para evitar que su información sea robada, Firefox no ha conectado con este sitio web.
Saber más...
Ir atrás
Avanzado
172.20.73.11 uses an invalid security certificate.
The certificate is not trusted because it is self-signed.
The certificate is only valid for ubuntu
Error code: SEC_ERROR_UNKNOWN_ISSUER
Añadir excepción...


Añadir excepción de seguridad
Está a punto de alterar cómo identifica Firefox este sitio.
Los bancos, tiendas y otros sitios públicos legítimos no le pedirán hacer esto.
Servidor
Dirección:
https://172.20.73.11/
Obtener certificado
Estado del certificado
Este sitio intenta identificarse a sí mismo con información no válida.
Ver...
Sitio erróneo
El certificado pertenece a un sitio diferente, lo que podría significar que alguien está intentando hacer pasar por este sitio.
Identidad desconocida
No se confía en el certificado porque no ha sido verificado como emitido por una autoridad confiable usando una firma segura.
Guardar esta excepción de manera permanente
Confirmar excepción de seguridad
Cancelar