

Seguridad en Apache



¿Qué es Apache?

El servidor HTTP Apache es un servidor web HTTP de código abierto, se puede utilizar para plataformas como Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, las cuales implementa el protocolo HTTP/1.1.

Lo podemos ver como un programa que nos permite acceder a las páginas web de un ordenador desde otro.

¿Por qué utilizar Apache?

- Es multiplataforma.
- Es configurable y de código abierto.
- Trabaja con una gran variedad de lenguajes de programación.
- Tiene archivos Log.
- Es fácil obtener soporte.

Directiva TraceEnable

Para realizar una demostración de estas directivas necesitaremos el comando "curl" con la opción "-I", la cual nos permitirá ver solo las cabeceras, sin necesidad de descargarnos la página, como podemos ver ahora:

```
lorenmanu@lorenmanuServer:~$ curl -I http://192.168.1.101
HTTP/1.1 200 OK
Date: Wed, 13 May 2015 22:10:30 GMT
Server: Apache/2.2.22 (Ubuntu)
Last-Modified: Thu, 16 Apr 2015 13:57:35 GMT
ETag: "71cb-ce-513d7d88451c0"
Accept-Ranges: bytes
Content-Length: 206
Vary: Accept-Encoding
Content-Type: text/html
```

Figura 2.1: Visualización de la orden curl.

- La siguiente la línea "HTTP/1.1 200 OK", puede ser usada por atacante, en concreto "Cross Site Scripting" o "XSS".
- Para desactivarla nos tendríamos que dirigir al archivo "/etc/apache2/conf.d/security" y ponemos:

```
TraceEnable Off
```

- De tal manera que el resultado sería: "HTTP/1.1".

Directiva ServerTokens

Otra directiva es ServerTokens, la cual indica en el campo Server de las cabeceras de las respuestas si se especifica el sistema operativo del servidor así como información sobre los módulos compilados. Esta información puede ser usada por un atacante, para solucionarlo Apache ofrece diferentes soluciones, se recomienda utilizar la siguiente, ya que en ella solo se especifica el servidor que se usa, por lo que en el archivo :

`"/etc/apache2/conf.d/security"`

le pondremos a la directiva ServerTokens el siguiente valor:

```
ServerTokens Prod
```

De tal manera que saldría: "Server: Apache".

Directiva DocumentRoot

Esta directiva especifica el directorio desde el cuál httpd servirá los ficheros. A menos que especifique alguna otra equivalencia mediante una directiva Alias, el servidor añade la ruta de la URL solicitada a este directorio para construir la ruta del documento a servir. Ejemplo:

```
"DocumentRoot /usr/web"
```

esto quiere decir que una petición de acceso a

```
"http://www.my.host.com/index.html"
```

se refiere a

```
"/usr/web/index.html"
```

en el sistema de ficheros.

Podemos cambiar el valor de esta directiva desde:

```
"/etc/apache2/apache2.conf"
```

Directiva Alias

Una directiva similar sería Alias, la cual nos da la capacidad de acceder a un directorio mediante un nombre, es decir, si tuviéramos un directorio denominado `"/var/www/sitio5"`, así pues si pusiéramos al final de la directiva Directory:

```
#<Directory />
#     AllowOverride None
#     Order Deny,Allow
#     Deny from all
#</Directory>

Alias /prueba1 "/var/www/index.html"
```

Podríamos acceder a `index.html` de dos formas tal y como mostramos en las siguientes dos figuras:

```
lorenmanu@lorenmanuServer:~$ curl http://192.168.1.100/index.html
<html><body><h1>It works!</h1>
<p>This is the default web page for this server.</p>
<p>The web server software is running but no content has been added, yet.</p>
<p> Soy la maquina original.
</body></html>
lorenmanu@lorenmanuServer:~$
```

Figura 3.2: Primera demostración de la directiva alias.

```
lorenmanu@lorenmanuServer:~$ curl http://192.168.1.100/prueba1
<html><body><h1>It works!</h1>
<p>This is the default web page for this server.</p>
<p>The web server software is running but no content has been added, yet.</p>
<p> Soy la maquina original.
</body></html>
lorenmanu@lorenmanuServer:~$
```

Figura 3.3: Segunda demostración de la directiva alias.

Podemos cambiar el valor de esta directiva en el archivo:

`"/etc/apache2/conf.d/security"`

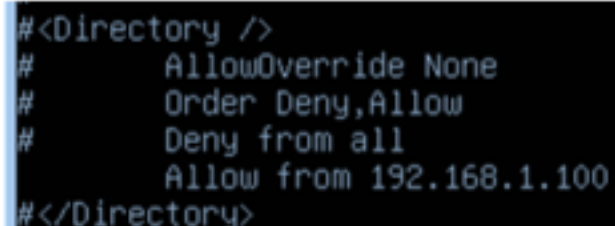
Directiva Directory

Nos ofrece un grupo de directivas que se aplicarán solamente al directorio del sistema de ficheros especificado y a sus subdirectorios. Su sintaxis es la siguiente:

```
"<Directory /usr/local/httpd/htdocs>  
Options Indexes FollowSymLinks  
</Directory>"
```

Unas de las opciones que nos ofrece y que nos ha sido muy importante destacar es la de permitir a unas IPs determinadas acceder a la ruta de archivos especificada, es decir, establecemos un firewall. Para accederemos al archivo de configuración:

```
"/etc/apache2/conf.d/security"  
Y pondremos lo siguiente:
```



```
#<Directory />  
#     AllowOverride None  
#     Order Deny,Allow  
#     Deny from all  
#     Allow from 192.168.1.100  
#</Directory>
```

Figura 4.1: Visualización del valor de la directiva.

Demostración Directory

En el cual estamos indicando que niegue todos los equipos y que permita solo a los que tiene la IP 192.168.1.100.

Así pues podemos ver que si el equipo tiene la dirección IP 192.168.1.100 no accederá a la página web:

```
eth1    Link encap:Ethernet  direcciónHW 08:00:27:41:7a:40
        Direc. inet:192.168.1.100  Difus.:192.168.1.255  Másc:255.255.255.0
        Dirección inet6: fe80::a00:27ff:fe41:7a40/64  Alcance:Enlace
        ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST  MTU:1500  Métrica:1
        Paquetes RX:129 errores:0 perdidos:0 overruns:0 frame:0
        Paquetes TX:106 errores:0 perdidos:0 overruns:0 carrier:0
        colisiones:0 long.colatx:1000
        Bytes RX:18519 (18.5 KB)  TX bytes:16508 (16.5 KB)
```

Figura 4.2: Demostración paso 1.

```
lorenmanu@lorenmanuServer:~$ curl http://192.168.1.100
<html><body><h1>It works!</h1>
<p>This is the default web page for this server.</p>
<p>The web server software is running but no content has been added, yet.</p>
<p> Soy la maquina original.
</body></html>
lorenmanu@lorenmanuServer:~$ _
```

Figura 4.3: Demostración paso 2.

```
lorenmanu@lorenmanuServer:~$ curl http://192.168.1.100
curl: (7) couldn't connect to host
lorenmanu@lorenmanuServer:~$
```

Figura 4.4: Demostración paso 3.

Otras directivas.

Otras directivas que se pueden destacar para evitar Ataques DDOS son las siguientes:

"Timeout": Esta directiva indica el tiempo que el servidor esperará para que un evento termine antes de fallar. Su valor por defecto es de 300 segundos. Es bueno mantener este valor lo más bajo posible si nuestro sitio es constantemente sometido a este tipo de ataque.

"MaxClients": Esta directiva nos permite configurar el número de conexiones que vamos a permitir simultáneamente. Las conexiones nuevas serán puestas en cola a partir del límite que configuremos.

"KeepAliveTimeout": Es el tiempo máximo que el servidor esperará para un requerimiento posterior antes de cerrar la conexión.

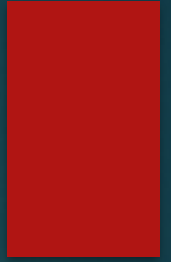
"LimitRequestFields": Nos ayuda a limitar el número de solicitudes de encabezados HTTP que aceptaremos de un cliente. Su valor por defecto es 100. Es recomendable que reduzcamos el valor si constantemente somos blancos de ataques DDOS.

Crear un certificado SSL de firma propia con OpenSSL y Apache HTTP Server.



- ▶ Un certificado SSL es un certificado digital utilizado para cifrar la información de un sitio y crear conexión segura.
- ▶ Este certificado es proporcionado por un proveedor autorizado (Digicert, Comodo, Verisign. . .) y es enviado por el servidor con quien estamos establecido la conexión al cliente.
- ▶ Apache dispone de un módulo llamado mod-ssl el cual provee soporte para SSL versiones 2 y 3 y TLS versión 1.
- ▶ OpenSSL crea certificados que otorgan el mismo nivel de cifrado como cualquier proveedor autorizado.

Funcionamiento básico



- ▶ 1 Solicitud de establecimiento de una conexión segura por SSL
- ▶ 2 Presentación del certificado
- Comprobaciones realizadas en el certificado en esta fase:
 - Validez
 - Firma por un tercero de confianza
- ▶ Transmisión de una clave de encriptación única codificada con la clave pública del servidor.
- ▶ Se Desencripta la clave por el servidor utilizando su clave privada.
 - Establecimiento de la conexión segura.



Pasos para la creación de un certificado SSL



- ▶ Activamos el módulo con el siguiente comando:
`# a2enmod ssl`
- ▶ Después reiniciamos el servidor apache para efectuar los cambios:
`# service apache2 restart`
- ▶ Creamos nuevo directorio donde vamos a almacenar la clave del servidor y el certificado:
`# mkdir /etc/apache2/ssl`
- ▶ Creamos un certificado SSL autofirmado con el openssl y la clave del servidor que lo protege, y colocamos ambos en el nuevo directorio.
`# openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/apache2/ssl/apache.key -out /etc/apache2/ssl/apache.crt`

Pasos para la creación de un certificado SSL

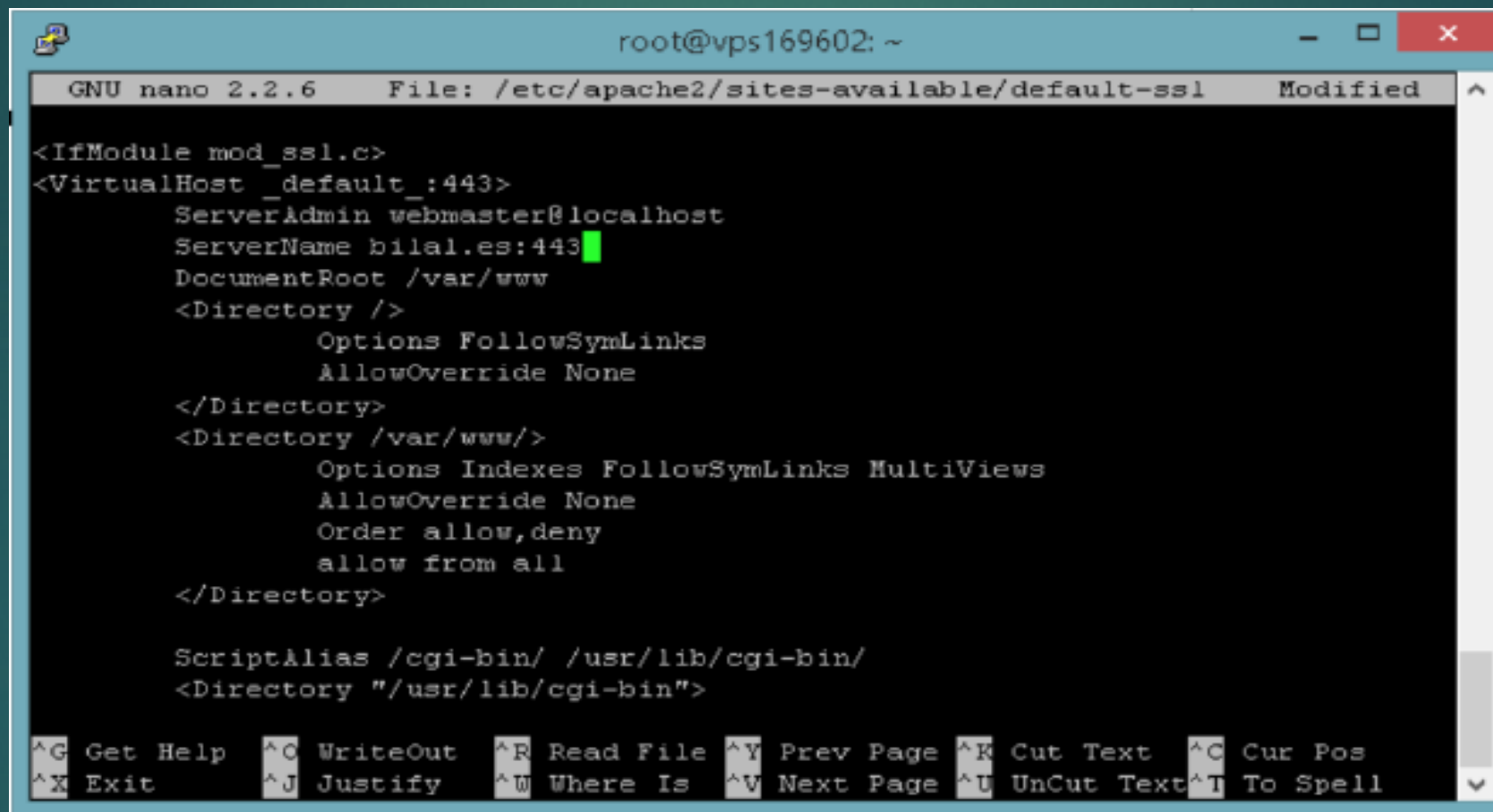
```
root@vps169602: ~  
ut /etc/apache2/ssl/apache.key -out /etc/apache2/ssl/apache.crt  
Generating a 2048 bit RSA private key  
.....+++  
.....+++  
writing new private key to '/etc/apache2/ssl/apache.key'  
-----  
You are about to be asked to enter information that will be incorporated  
into your 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]:GR  
Locality Name (eg, city) []:Granada  
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Bilal  
Organizational Unit Name (eg, section) []:IT  
Common Name (e.g. server FQDN or YOUR name) []:bilal.es  
Email Address []:yesfi.bilal@gmail.com  
root@vps169602:~# ^C  
root@vps169602:~#
```

Instalación del certificado



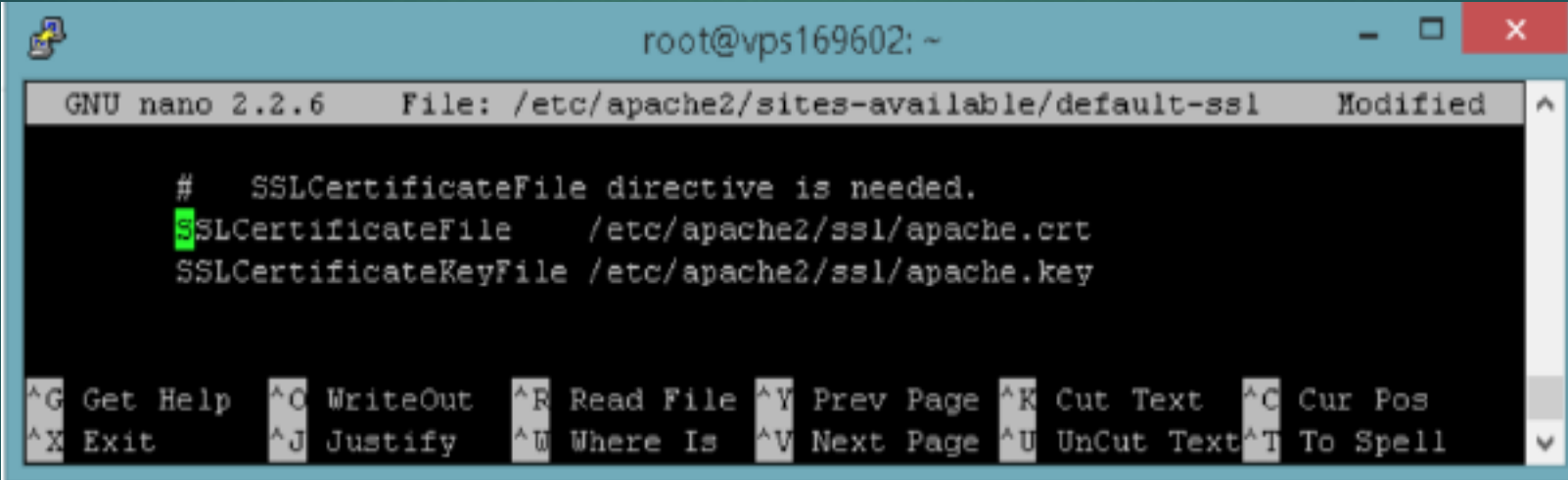
- ▶ En este paso hay que configurar los hosts virtuales para mostrar el nuevo certificado.
- ▶ Abrimos el archivo de configuración:
`# nano /etc/apache2/sites-available/default-ssl`
- ▶ Añadir dentro de la sección `<VirtualHost _default_:443>` la siguiente línea: `ServerName example.com:443`
- ▶ Luego modifica estos dos campos:
SSLCertificateFile con `/etc/apache2/ssl/apache.crt`
SSLCertificateKeyFile con `/etc/apache2/ssl/apache.key`

Instalación del certificado



```
root@vps169602: ~  
GNU nano 2.2.6 File: /etc/apache2/sites-available/default-ssl Modified  
<IfModule mod_ssl.c>  
<VirtualHost _default_:443>  
    ServerAdmin webmaster@localhost  
    ServerName bilal.es:443  
    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>  
  
    ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/  
    <Directory "/usr/lib/cgi-bin">  
  
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos  
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```


Instalación del certificado



The screenshot shows a terminal window titled 'root@vps169602: ~'. Inside, the GNU nano 2.2.6 editor is open, editing the file '/etc/apache2/sites-available/default-ssl'. The file content is as follows:

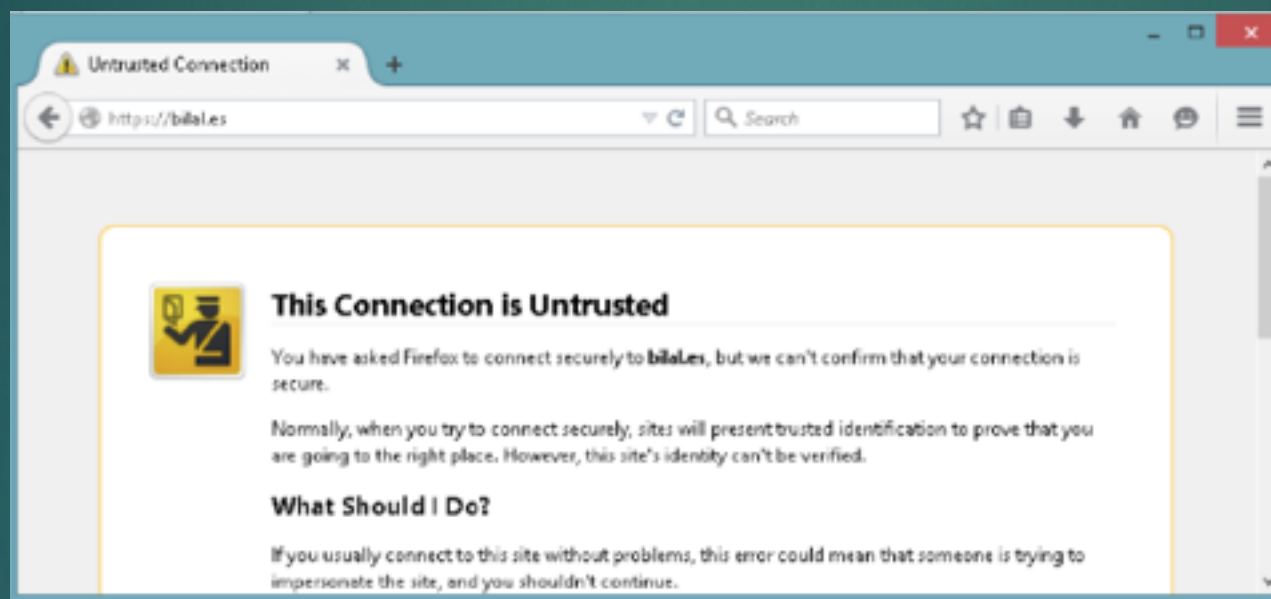
```
# SSLCertificateFile directive is needed.  
SSLCertificateFile /etc/apache2/ssl/apache.crt  
SSLCertificateKeyFile /etc/apache2/ssl/apache.key
```

The 'S' in 'SSLCertificateFile' is highlighted in green. At the bottom of the terminal, a list of nano editor shortcuts is visible:

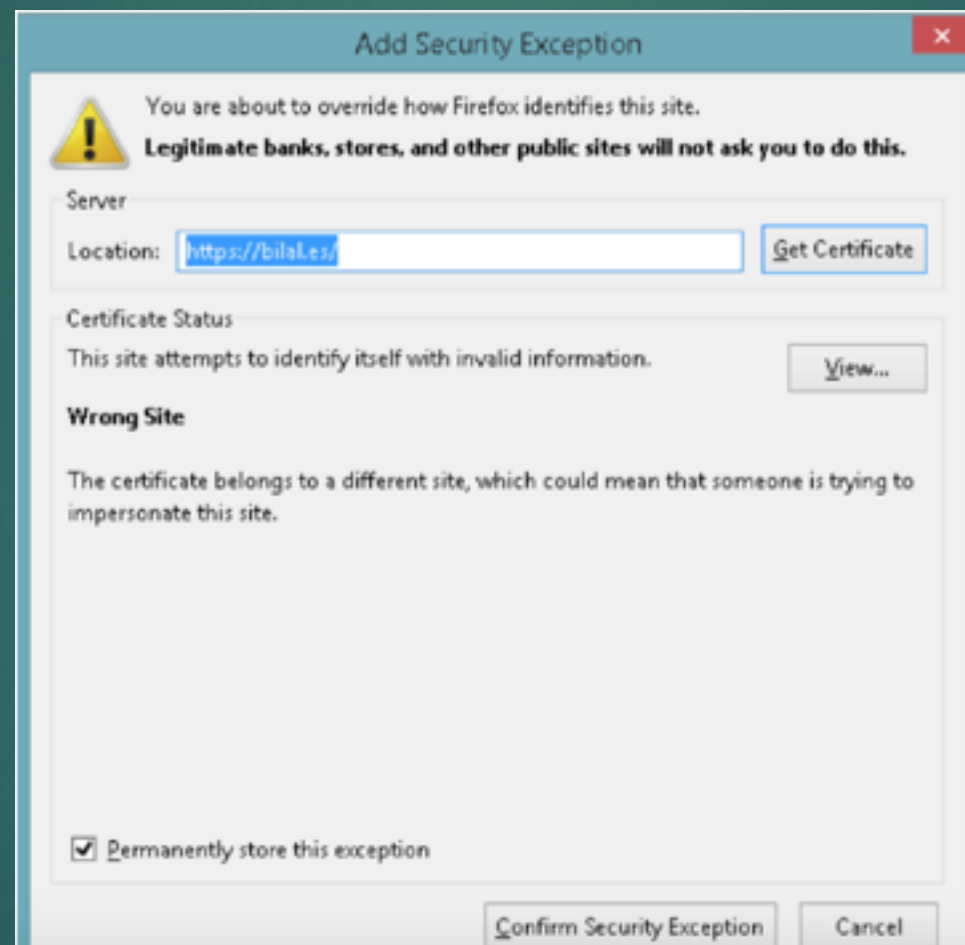
```
^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos  
^X Exit      ^J Justify    ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

- ▶ Activar el nuevo Virtual-host:
a2ensite default-ssl
- ▶ Finalmente, reiniciamos el apache:
service apache2 reload

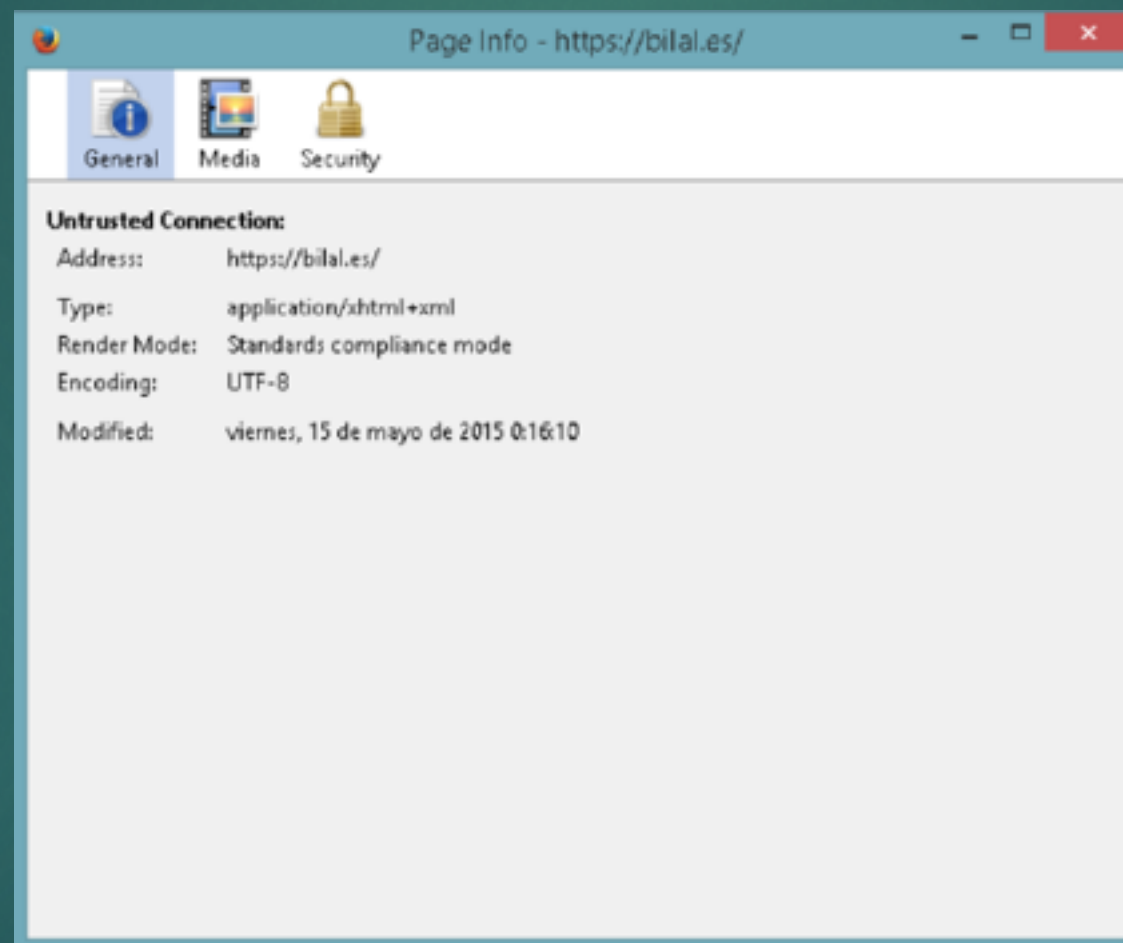
Verificación del certificado SSL desde un navegador



Verificación del certificado SSL desde un navegador



Verificación del certificado SSL desde un navegador



Captura de wireshark

50	3.895862000	172.20.80.83	92.222.19.192	TCP	62 49424-443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 SACK_PERM=1
51	3.896145000	172.20.80.83	92.222.19.192	TCP	62 49425-443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 SACK_PERM=1
56	3.947692000	172.20.80.83	92.222.19.192	TCP	54 49425-443 [ACK] Seq=1 Ack=1 Win=16616 Len=0
58	3.947849000	172.20.80.83	92.222.19.192	TCP	54 49424-443 [ACK] Seq=1 Ack=1 Win=16616 Len=0
59	3.949276000	172.20.80.83	92.222.19.192	TLSv1.2	253 Client Hello
60	3.949385000	172.20.80.83	92.222.19.192	TLSv1.2	253 Client Hello
71	4.008755000	172.20.80.83	92.222.19.192	TCP	54 49424-443 [ACK] Seq=200 Ack=1445 Win=16616 Len=0
82	4.133141000	172.20.80.83	92.222.19.192	TLSv1.2	180 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
90	4.155362000	172.20.80.83	92.222.19.192	TCP	54 49424-443 [FIN, ACK] Seq=326 Ack=1445 Win=16616 Len=0
92	4.203898000	172.20.80.83	92.222.19.192	TCP	54 49424-443 [ACK] Seq=327 Ack=1719 Win=16342 Len=0
94	4.207702000	172.20.80.83	92.222.19.192	TCP	54 49424-443 [ACK] Seq=327 Ack=1720 Win=16342 Len=0
96	4.251926000	172.20.80.83	92.222.19.192	TLSv1.2	253 [TCP Retransmission] Client Hello

- En la captura de wireshark vemos que la conexión entre el cliente y el servidor 92.222.19.192 se hace a través del puerto 433 y el uso de TLSv1.2.