



Este certificado SSL se puede usar para habilitar **HTTPS en Apache** para un entorno local o de pruebas. No está firmado por una entidad certificadora reconocida, por lo que los navegadores lo marcarán como "*no seguro*", pero es útil para desarrollo.

Requisitos:

- Este comando usa openssl versión 1.1.1 o superior, ya que incluye la opción -addext. Si se dispone de una versión más antigua, habría que usar un archivo de configuración adicional.

Explicación código:

```
sudo openssl req \
  -newkey rsa:2048 \
  -nodes \
  -keyout /etc/apache2/ssl/server.key \
  -x509 \
  -days 365 \
  -out /etc/apache2/ssl/server.crt \
  -subj "/C=US/ST=State/L=City/O=Organization/OU=Department/CN=localhost" \
  -addext "basicConstraints=CA:FALSE" \
  -addext "subjectAltName=DNS:localhost"
```

# Genera una nueva clave RSA de 2048 bits  
# No encripta la clave privada (sin contraseña)  
# Ruta donde se guarda la clave privada  
# Genera un certificado autofirmado (tipo X.509)  
# El certificado será válido por 365 días  
# Ruta donde se guarda el certificado  
# Datos del sujeto  
# Indica que este certificado no es una CA  
# Agrega un nombre alternativo (SAN) para localhost

**Nota:** Se puede usar un certificado de una autoridad certificadora como *Let's Encrypt* en lugar de un *autofirmado*.



## Configurar Apache para usar TLS

Editar el archivo de configuración SSL de Apache `default-ssl.conf`:

```
sudo mousepad /etc/apache2/sites-available/default-ssl.conf
```

Añadir las siguientes líneas:

```
<VirtualHost *:443>
  ServerAdmin admin@localhost
  ServerName localhost
  DocumentRoot /var/www/html

  SSLEngine on
  SSLCertificateFile /etc/apache2/ssl/server.crt
  SSLCertificateKeyFile /etc/apache2/ssl/server.key

  SSLProtocol all -SSLv3 -TLSv1 -TLSv1.1
  SSLCipherSuite HIGH:!aNULL:!MD5
</VirtualHost>
```

Guardar el archivo.

Apache **debe** soportar **TSL**, lo cual depende de la versión de Apache y OpenSSL. A partir de **Apache 2.4.38+** con **OpenSSL 1.1.1+**, TLS 1.3 es compatible ([Apache HTTPD Documentation](#)). SSLProtocol TLSv1.3

## Habilitar SSL y reiniciar Apache

Habilitar el módulo SSL y el sitio seguro:

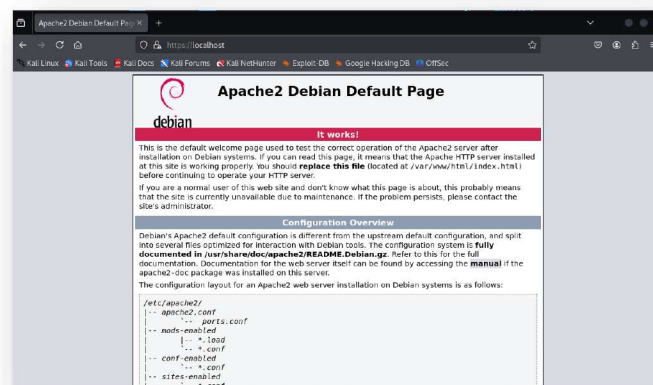
```
sudo a2enmod ssl
sudo a2ensite default-ssl
```

Reiniciar Apache para aplicar los cambios:

```
sudo systemctl restart apache2
```

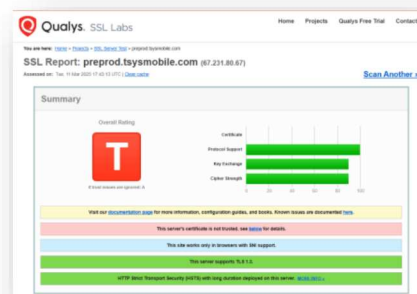
Abrimos navegador y lanzamos:

https://localhost



### Verificar la configuración con SSL Labs *(con dominio)*

Para asegurarse de que la configuración de TLS es segura, se puede comprobar el dominio en: SSL Labs Test <https://www.ssllabs.com/ssltest/>



## Mitigación de problemas comunes

### \* Deshabilitar versiones inseguras de TLS

Para evitar vulnerabilidades, en *default-ssl.conf* configurar:

```
SSLProtocol TLSv1.2 TLSv1.3
```

### \* Forzar HTTPS con HSTS

Añadir dentro del bloque *<VirtualHost \*:443>*:

```
Header always set Strict-Transport-Security "max-age=31536000; includeSubDomains; preload"
```

Habilitar los encabezados HTTP en Apache con:

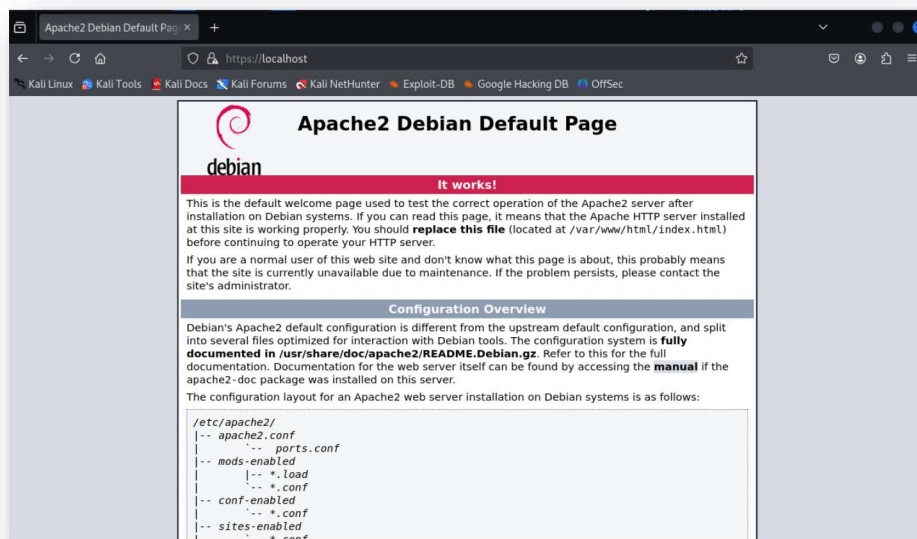
```
sudo a2enmod headers
sudo systemctl restart apache2
```

### \* Verificar la conexión HTTPS

Para comprobar si HTTPS funciona correctamente, acceder a:

```
https://localhost
```

Si se observa el *candado* en la barra de direcciones, TLS está activo.



## ¿Cómo eliminar la advertencia del candado? (Opcional)

Si solo trabajas en local, no hay problema en ignorar la advertencia. Pero si se quiere que el navegador lo reconozca como seguro sin advertencias, dado que Firefox solo permite importar certificados de CA en la pestaña "Authorities", se debe generar un certificado raíz de CA y luego firmar el certificado con él.

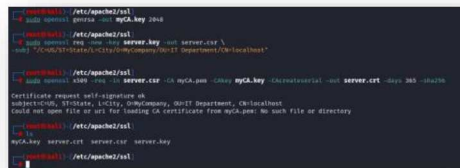
### 1. Crear un Certificado de Autoridad (CA)

Ejecutar estos comandos para generar una CA local:

```
mkdir -p /etc/apache2/ssl
cd /etc/apache2/ssl

# Generar la clave privada de la CA
sudo openssl genrsa -out myCA.key 2048

# Generar el certificado raíz de la CA (válido por 10 años)
sudo openssl req -x509 -new -nodes -key myCA.key -sha256 -days 3650 -out myCA.pem \
-subj "/C=US/ST=State/L=City/O=MyCompany/OU=IT Department/CN=My Local CA"
```



### 2. Firmar el Certificado SSL con la CA Local

Generar una clave para el servidor Apache:

```
sudo openssl genrsa -out server.key 2048
```

Crear una solicitud de firma (CSR Certificate Signing Request, es un archivo que contiene información sobre una entidad que solicita un certificado SSL/TLS):

```
sudo openssl req -new -key server.key -out server.csr \
-subj "/C=US/ST=State/L=City/O=MyCompany/OU=IT Department/CN=localhost"
```

Firmar el certificado con nuestra CA local:

```
sudo openssl x509 -req -in server.csr -CA myCA.pem -CAkey myCA.key -CAcreateserial -out
server.crt -days 365 -sha256
```

Ahora disponemos de un certificado **server.crt** firmado por nuestra CA **myCA.pem**.



```
(root@kali)-[/etc/apache2/ssl]
# sudo openssl genrsa -out server.key 2048

(root@kali)-[/etc/apache2/ssl]
# sudo openssl req -new -key server.key -out server.csr \
-subj "/C=US/ST=State/L=City/O=MyCompany/OU=IT Department/CN=localhost"

(root@kali)-[/etc/apache2/ssl]
# sudo openssl x509 -req -in server.csr -CA myCA.pem -CAkey myCA.key -CAcreateserial -out server.crt -days 365 -sha256
Certificate request self-signature ok
subject=C=US, ST=State, L=City, O=MyCompany, OU=IT Department, CN=localhost
Could not open file or uri for loading CA certificate from myCA.pem: No such file or directory

(root@kali)-[/etc/apache2/ssl]
#
```

### 3. Configurar Apache con el Nuevo Certificado

Asegurar de que Apache use los nuevos certificados. Para ello modificar:

```
sudo mousepad /etc/apache2/sites-available/default-ssl.conf
```

Cambiar las rutas para que apunten a:

```
SSLEngine on
SSLCertificateFile /etc/apache2/ssl/server.crt
SSLCertificateKeyFile /etc/apache2/ssl/server.key
SSLCertificateChainFile /etc/apache2/ssl/myCA.pem
```

Guardar y reiniciar Apache:

```
sudo systemctl restart apache2
```

### 4. Importar la CA en Firefox

Importar myCA.pem en la pestaña "Authorities" de Firefox:

1. Abrir Firefox e ir a *about:preferences#privacy*
2. En *Certificados* y seleccionar *Ver certificados*
3. En la pestaña *Autoridades* y seleccionar *Importar...*
4. Seleccionar */etc/apache2/ssl/myCA.pem*
5. Marcar la casilla *"Confiar en esta CA para identificar sitios web"*
6. Guardar los cambios.

Firefox confiará en los certificados firmados por esta CA, y la advertencia desaparecerá.

## ANEXO I

En sistemas más actualizados, se puede reemplazar:

```
SSLProtocol TLSv1.2 TLSv1.3
SSLCipherSuite HIGH:!aNULL:!MD5
```

por:

```
SSLOpenSSLConfCmd MinProtocol TLSv1.3
SSLOpenSSLConfCmd CipherString DEFAULT@SECLEVEL=2
```

Esto usa los ajustes por defecto del sistema con un buen nivel de seguridad (SECLEVEL=2 es el mínimo recomendado para producción).

### Notas importantes:

- Esta opción requiere **Apache 2.4.43 o superior** y **OpenSSL 1.1.1 o superior**.
- Si se usa `SSLOpenSSLConfCmd`, es **preferible no usar `SSLProtocol`**, para evitar conflictos.
- Se puede usar también `SSLOpenSSLConfCmd CipherString` para definir el conjunto de cifrados (similar a `SSLCipherSuite` pero más moderno y compatible con OpenSSL 1.1.1+ y 3.0+).

## ANEXO II

Se puede **verificar de manera local** que la configuración **TLS está funcionando correctamente**, especialmente útil cuando:

- No se dispone de un dominio público.
- Se está trabajando en un entorno de desarrollo o laboratorio.
- Se quiere confirmar que **TLS 1.3 está habilitado y operativo** antes de poner el servidor en producción.

```
openssl s_client -connect localhost:443 -tls1_3
```

Este comando:

- Intenta establecer una conexión TLS específicamente con la versión **1.3**.
- Muestra un resumen de la negociación TLS, incluyendo:
  - La versión del protocolo usada.
  - El certificado presentado.
  - El conjunto de cifrado negociado.

Si se obtiene en la salida:

```
Protocol    : TLSv1.3
```

Cipher : TLS\_AES\_256\_GCM\_SHA384

... entonces **TLS 1.3** está activo y funcionando.

```
nmap --script ssl-enum-ciphers -p 443 localhost
```

Este comando usa `nmap` para hacer un escaneo y **enumerar todas las versiones de TLS y conjuntos de cifrado que el servidor acepta**.

Requiere tener el paquete `nmap` instalado: `sudo apt install nmap`

En la salida se obtendrá algo similar a:

```
| ssl-enum-ciphers:
|   TLSv1.2:
|     ciphers:
|       TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
|   TLSv1.3:
|     ciphers:
|       TLS_AES_256_GCM_SHA384
|_  least strength: A
```

Esto confirma que el servidor acepta solo TLSv1.2 y TLSv1.3 (*si se configuró correctamente la exclusión de versiones antiguas*).

## Referencias

- **RFC 8446** – *The Transport Layer Security (TLS) Protocol Version 1.3*, IETF:  
<https://datatracker.ietf.org/doc/html/rfc8446>
- Mozilla's SSL Configuration Generator (muy útil para hardening de Apache):  
<https://ssl-config.mozilla.org/>
- Apache HTTPD TLS/SSL How-To  
[https://httpd.apache.org/docs/2.4/ssl/ssl\\_howto.html](https://httpd.apache.org/docs/2.4/ssl/ssl_howto.html)