

Unit 2. Task 3: SSL module activation and creation of a secure Apache server

Javier Aguilera Aguilera 2º CFGS DAW

Contents

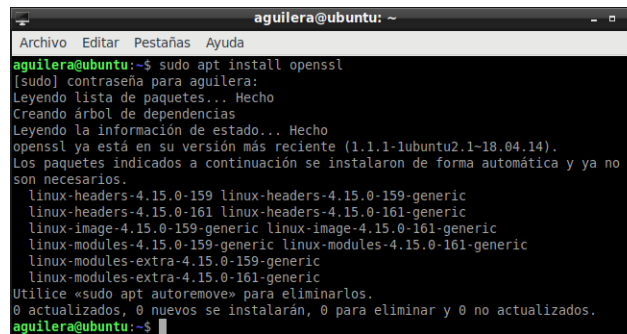
1	Installing openssl	3
2	Setting up the server key	3
2.1	Writing server key	3
3	Naming our server	4
4	Setting up the certificate duration	4
5	Making ssl directory	5
6	Enable SSL module	5
6.1	Testing if port 443 is opened	7
7	Reboot default-ssl and apache2	8
8	Testing	8

1 Installing openssl

In this task we are going to activate de SSL module in order to create a secure Apache server.

First of all, we will make sure whether SSL module is already installed in our system or not. Typing at the terminal

```
$ sudo apt install openssl
```



```
aguilera@ubuntu: ~  
Archivo  Editar  Pestañas  Ayuda  
aguilera@ubuntu:~$ sudo apt install openssl  
[sudo] contraseña para aguilera:  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
openssl ya está en su versión más reciente (1.1.1-1ubuntu2.1-18.04.14).  
Los paquetes indicados a continuación se instalaron de forma automática y ya no  
son necesarios.  
linux-headers-4.15.0-159 linux-headers-4.15.0-159-generic  
linux-headers-4.15.0-161 linux-headers-4.15.0-161-generic  
linux-image-4.15.0-159-generic linux-image-4.15.0-161-generic  
linux-modules-4.15.0-159-generic linux-modules-4.15.0-161-generic  
linux-modules-extra-4.15.0-159-generic  
linux-modules-extra-4.15.0-161-generic  
Utilice «sudo apt autoremove» para eliminarlos.  
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.  
aguilera@ubuntu:~$
```

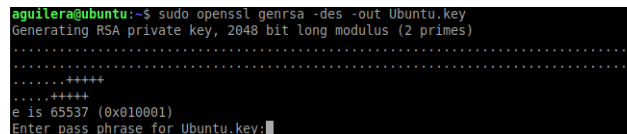
Figure 1: In my case, it is already installed

2 Setting up the server key

Then, we are going to to set up the server key by the command

```
$ sudo openssl genrsa -des -out server_key_name
```

Where server_key_name is the name of your server key. In my case, I will call it Ubuntu.key.



```
aguilera@ubuntu:~$ sudo openssl genrsa -des -out Ubuntu.key  
Generating RSA private key, 2048 bit long modulus (2 primes)  
.....+++++  
.....+++++  
e is 65537 (0x010001)  
Enter pass phrase for Ubuntu.key:
```

Figure 2: It ask for a passphrase

From now on, every time that the server_key_name is required I will write Ubuntu.key.

2.1 Writing server key

The following instruction to execute is

```
$ sudo openssl rsa -in Ubuntu.key -out Ubuntu.key
```

It replace the old server key with the new one we just defined.

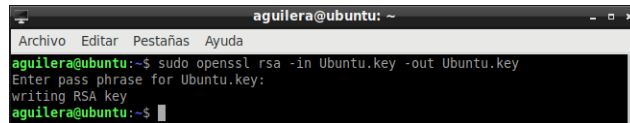


Figure 3: It ask for the previos pass phrase

3 Naming our server

Then, we have to write in the command line

```
$ sudo openssl req -new -key Ubuntu.key  
-out server_name
```

Where server_name is the name of your server. In my case, I will call it Secury.csr.

From now on, every time that the server_name is required I will write Secury.csr.

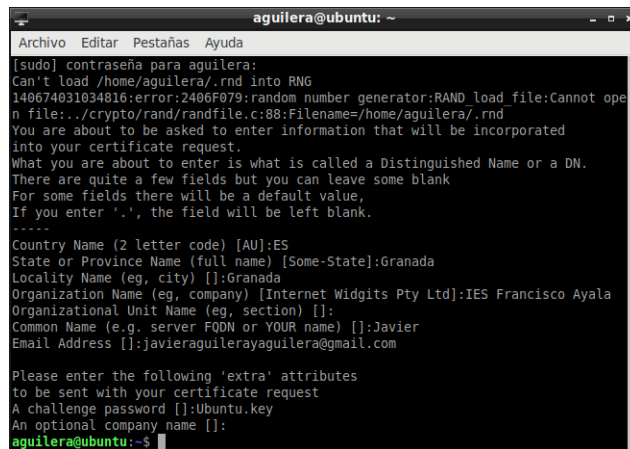


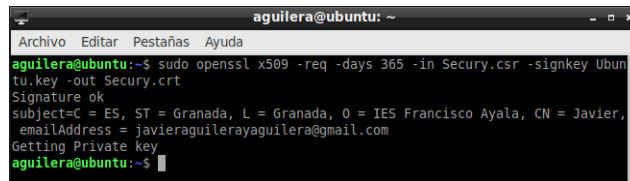
Figure 4: We need to fill some fields of required information

4 Setting up the certificate duration

Execute the command:

```
$ sudo openssl x509 -req -days 365 -in  
Secury.csr -signkey Ubuntu.key -out Secury.crt
```

365 is the duration equivalent a complete year.



```
aguilera@ubuntu: ~  
aguilera@ubuntu:~$ sudo openssl x509 -req -days 365 -in Secury.csr -signkey Ubuntu.key -out Secury.crt  
Signature ok  
subject=C = ES, ST = Granada, L = Granada, O = IES Francisco Ayala, CN = Javier,  
emailAddress = javieraguilerayaguilera@gmail.com  
Getting Private key  
aguilera@ubuntu:~$
```

Figure 5: Setting up the certificate duration

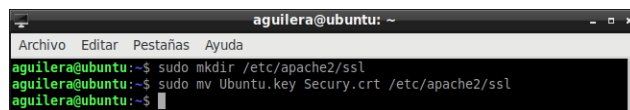
5 Making ssl directory

We are going to create a directory in order to store the server information set before. We have to write in the terminal

```
$ sudo mkdir /etc/apache2/ssl
```

Then, execute

```
$ sudo mv Ubuntu.key Secury.crt /etc/apache2/ssl
```



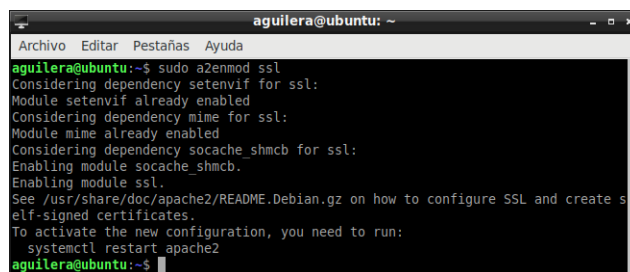
```
aguilera@ubuntu: ~  
aguilera@ubuntu:~$ sudo mkdir /etc/apache2/ssl  
aguilera@ubuntu:~$ sudo mv Ubuntu.key Secury.crt /etc/apache2/ssl  
aguilera@ubuntu:~$
```

Figure 6: Command summary

6 Enable SSL module

If we have right done all previous steps, it is time to enable the SSL module

```
$ sudo a2enmod ssl
```



```
aguilera@ubuntu: ~  
aguilera@ubuntu:~$ 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:  
systemctl restart apache2  
aguilera@ubuntu:~$
```

Figure 7: SSL enabled

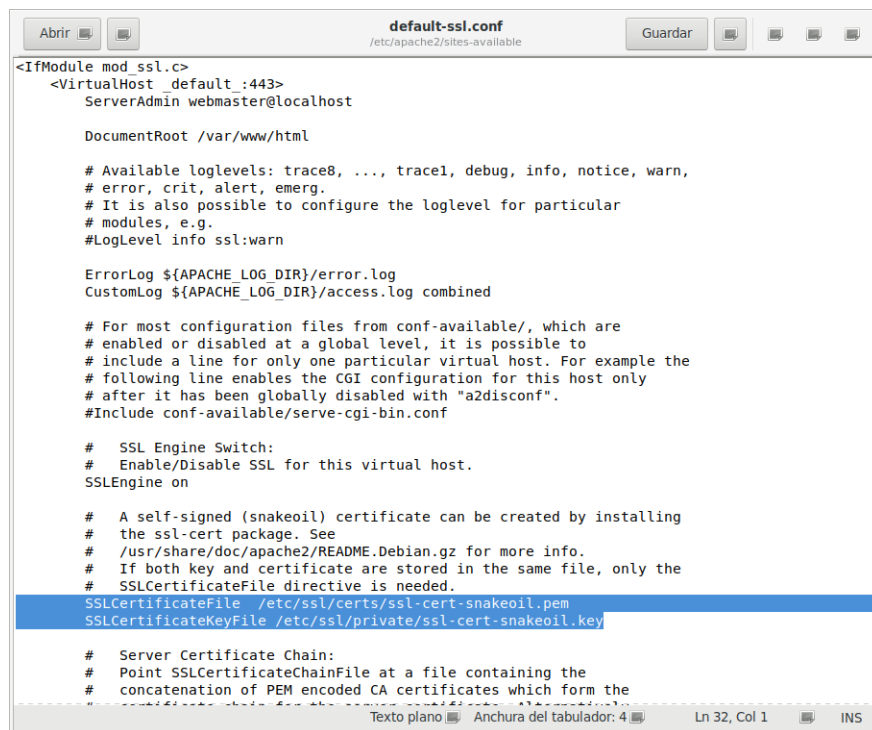
Finally, it is compulsory to modify `/etc/apache2/sites-available/default-ssl.conf`. Once you have opened it, you have to search the lines which contains “SSLCer-

tificateFile” and “SSLCertificateKeyFile” and you have to replace them with the followings:

- SSLCertificateFile /etc/apache2/ssl/Secury.crt
- SSLCertificateKeyFile /etc/apache2/ssl/Ubuntu.key

A valid option is to comment those lines with an “#” which allow to restore the original parameters

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



```
<!--
# To be able to use the functionality of a module which was not
# included in the base configuration, you have to place corresponding
# 'LoadModule' lines at this location so the functions can be
# loaded. It is good practice to put all loaded modules in one
# 'LoadModule' line as shown below.

# Load the ssl module.
LoadModule ssl_module modules/ssl.so

<IfModule mod_ssl.c>
<VirtualHost default *:443>
    ServerAdmin webmaster@localhost

    DocumentRoot /var/www/html

    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf

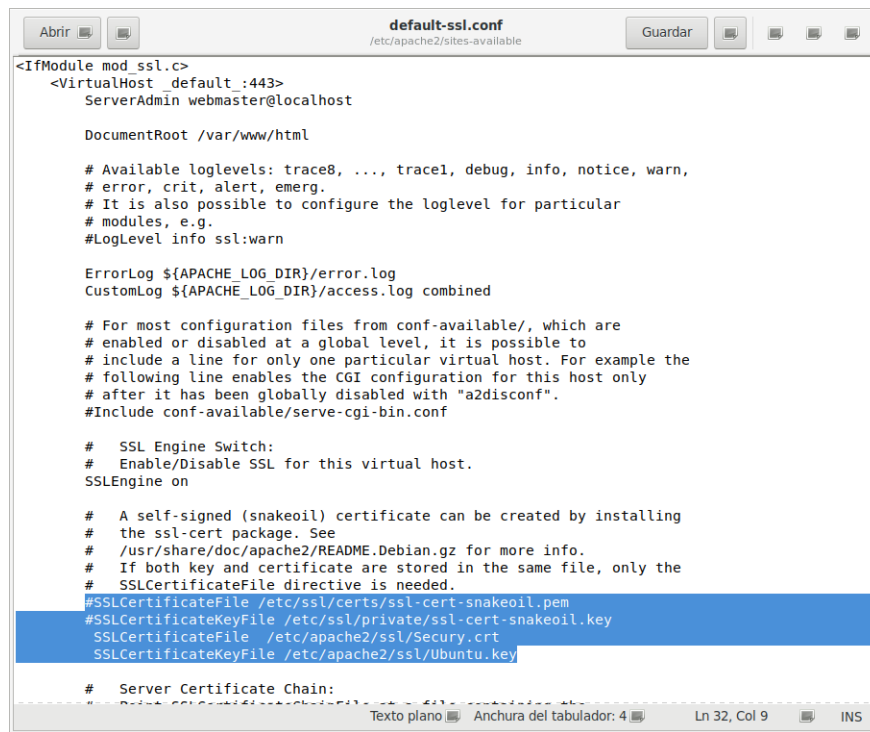
    #
    # SSL Engine Switch:
    # Enable/Disable SSL for this virtual host.
    SSLEngine on

    #
    # A self-signed (snakeoil) certificate can be created by installing
    # the ssl-cert package. See
    # /usr/share/doc/apache2/README.Debian.gz for more info.
    # If both key and certificate are stored in the same file, only the
    # SSLCertificateFile directive is needed.
    SSLCertificateFile /etc/ssl/certs/ssl-cert-snakeoil.pem
    SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key

    #
    # Server Certificate Chain:
    # Point SSLCertificateChainFile at a file containing the
    # concatenation of PEM encoded CA certificates which form the
    # certificate chain for the server certificate.
    #
    # SSLCertificateChainFile /

</VirtualHost>
</IfModule>-->
```

Figure 8: default-ssl.conf before modifications



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

<IfModule mod_ssl.c>
<VirtualHost default :443>
    ServerAdmin webmaster@localhost

    DocumentRoot /var/www/html

    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf

    #
    # SSL Engine Switch:
    # Enable/Disable SSL for this virtual host.
    SSLEngine on

    #
    # A self-signed (snakeoil) certificate can be created by installing
    # the ssl-cert package. See
    # /usr/share/doc/apache2/README.Debian.gz for more info.
    # If both key and certificate are stored in the same file, only the
    # SSLCertificateFile directive is needed.
    #SSLCertificateFile /etc/ssl/certs/ssl-cert-snakeoil.pem
    #SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
    SSLCertificateFile /etc/apache2/ssl/Secury.crt
    SSLCertificateKeyFile /etc/apache2/ssl/Ubuntu.key

    #
    # Server Certificate Chain:
    # ...

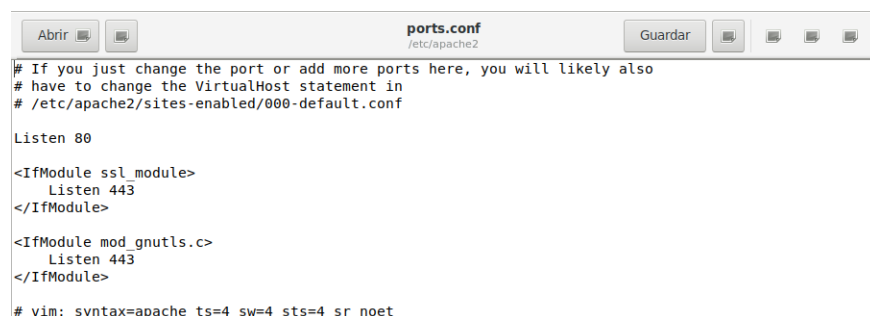
    # vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Figure 9: default-ssl.conf after modifications

6.1 Testing if port 443 is opened

In order to verify if port 443 is opened, we have to open with a text editor the *ports.conf* file (*/etc/apache2/ports.conf*), in my case with Gedit:

```
$ sudo gedit /etc/apache2/ports.conf
```



```
ports.conf
/etc/apache2

# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 80

<IfModule ssl_module>
    Listen 443
</IfModule>

<IfModule mod_gnutls.c>
    Listen 443
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Figure 10: As we can see, Listen 443 is uncommented, so it will work

7 Reboot default-ssl and apache2

Using the commands

```
$ sudo a2ensite default-ssl
```

```
$ sudo systemctl reload apache2
```

8 Testing

Last step is to check if we have made our configuration correctly. Open your favorite browser and search for “https://localhost” and a similar page to this is going to appear:

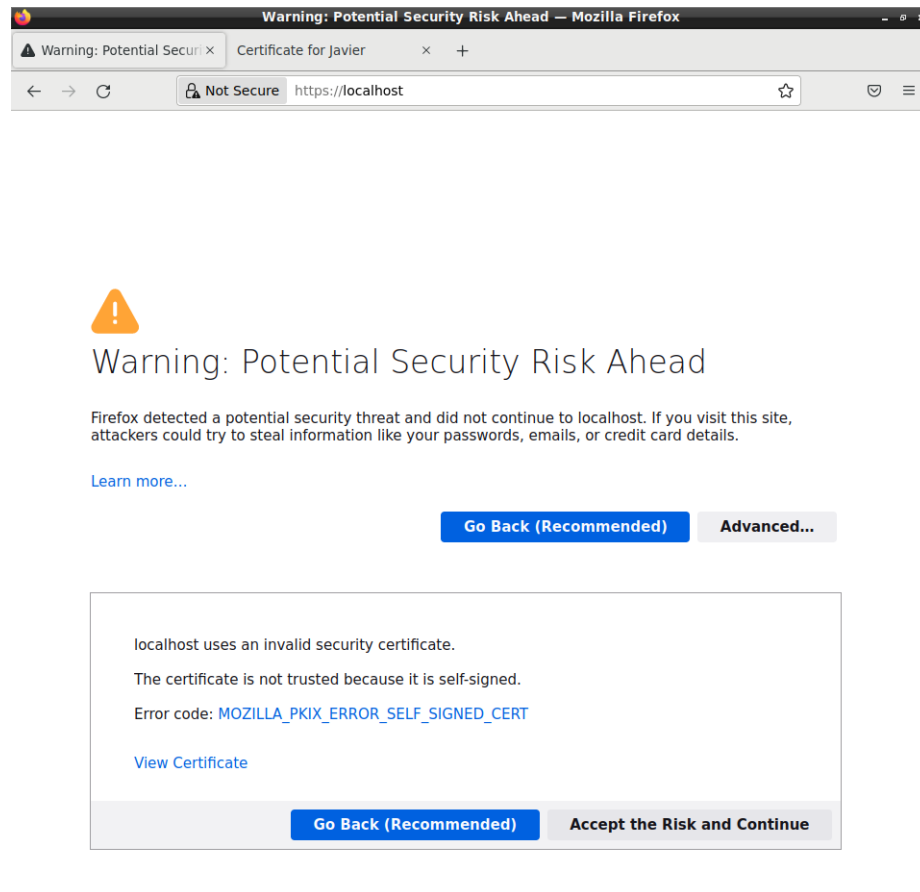


Figure 11: It says that it is not secure, let's see in *View Certificate*

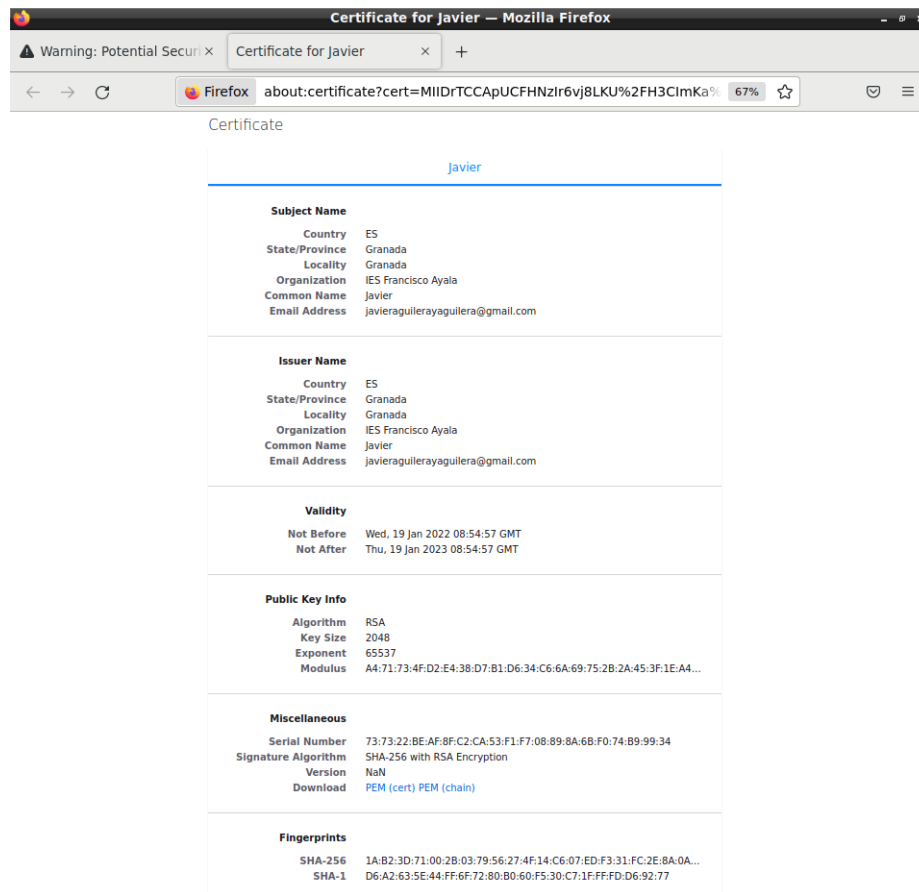


Figure 12: Our certificate information

To access localhost, we have to “Accept the Risk and Continue”. It is a security measure enabled by default because our certificate has not been validated by an official organism.

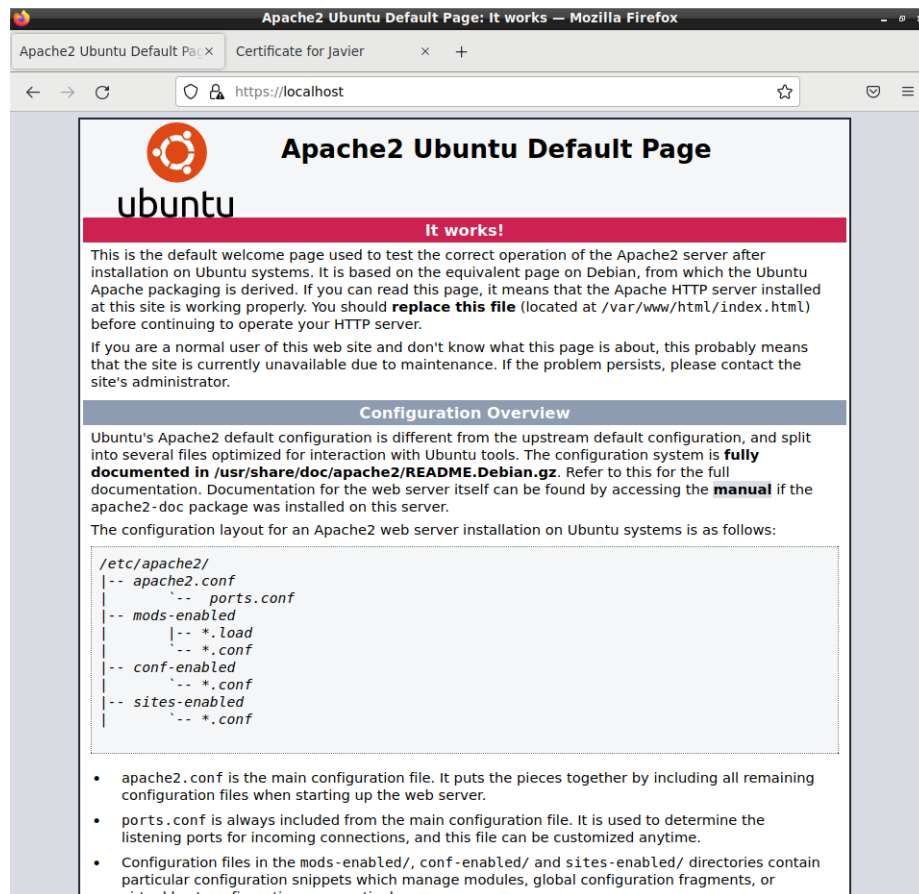


Figure 13: It works!

References

- [1] <https://www.youtube.com/watch?v=E5WFZWj070Y>