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

Javier Aguilera Aguilera 2º CFGS DAW

Contents

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
```

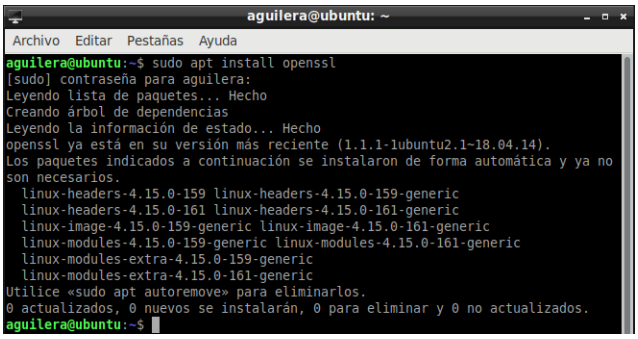

Figure 1: In my case, it is already installed

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.

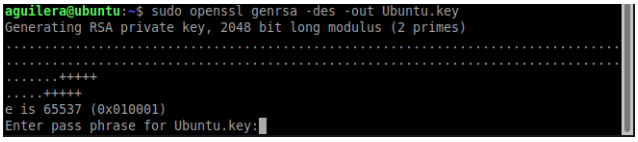

Figure 2: It ask for a passphrase

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.

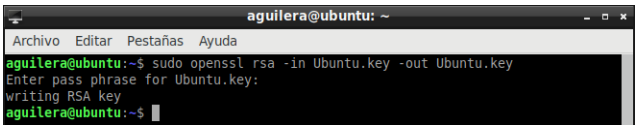

Figure 3: It ask for the previos pass phrase

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.

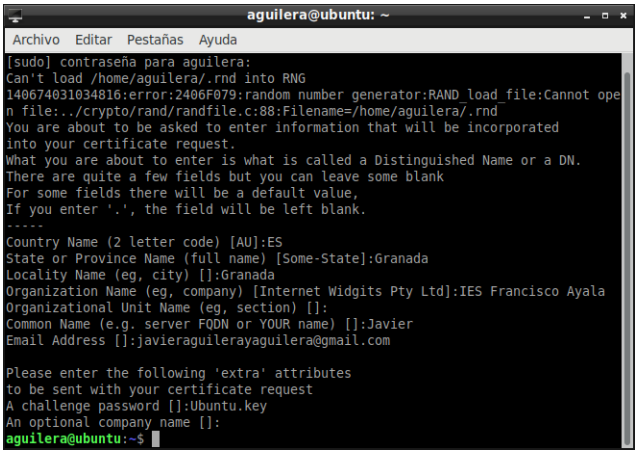

Figure 4: We need to fill some fields of required information

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.

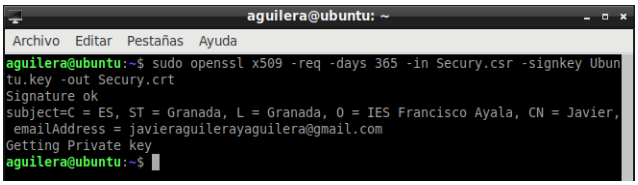

Figure 5: Setting up the certificate duration

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
```

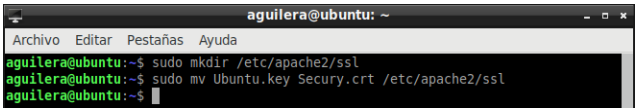

Figure 6: Command summary

6 Enable SSL module

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

```
$ sudo a2enmod ssl
```

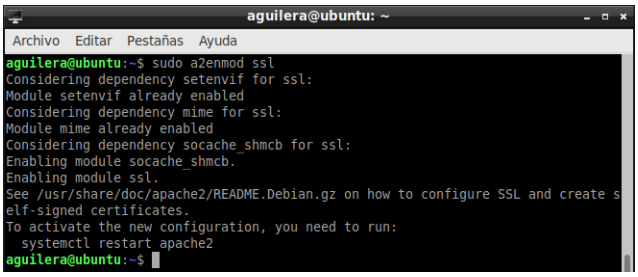

Figure 7: SSL enabled

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
```

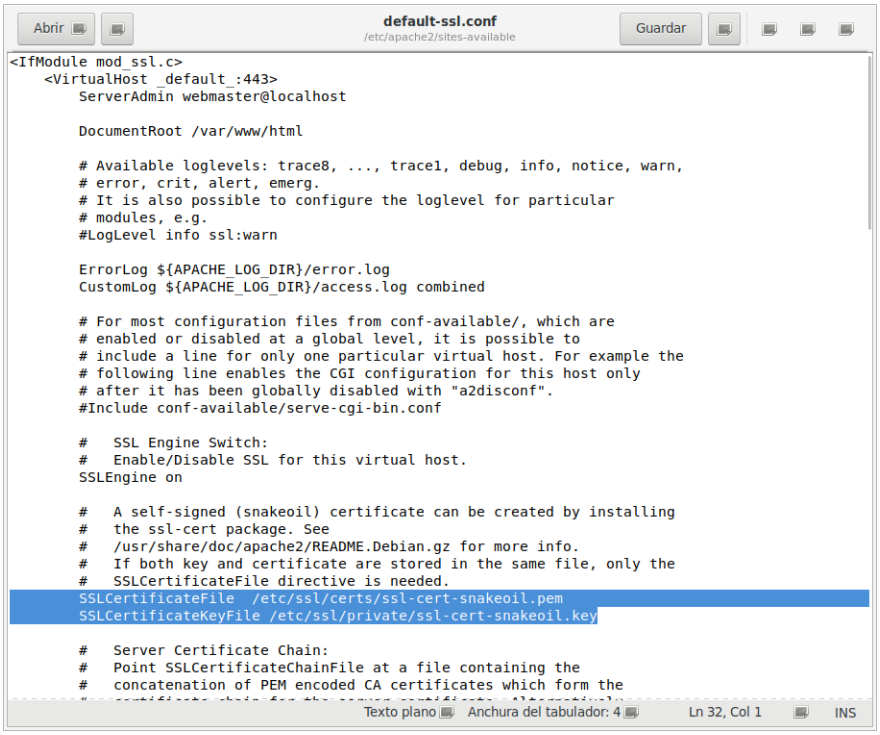

Figure 8: default-ssl.conf before modifications

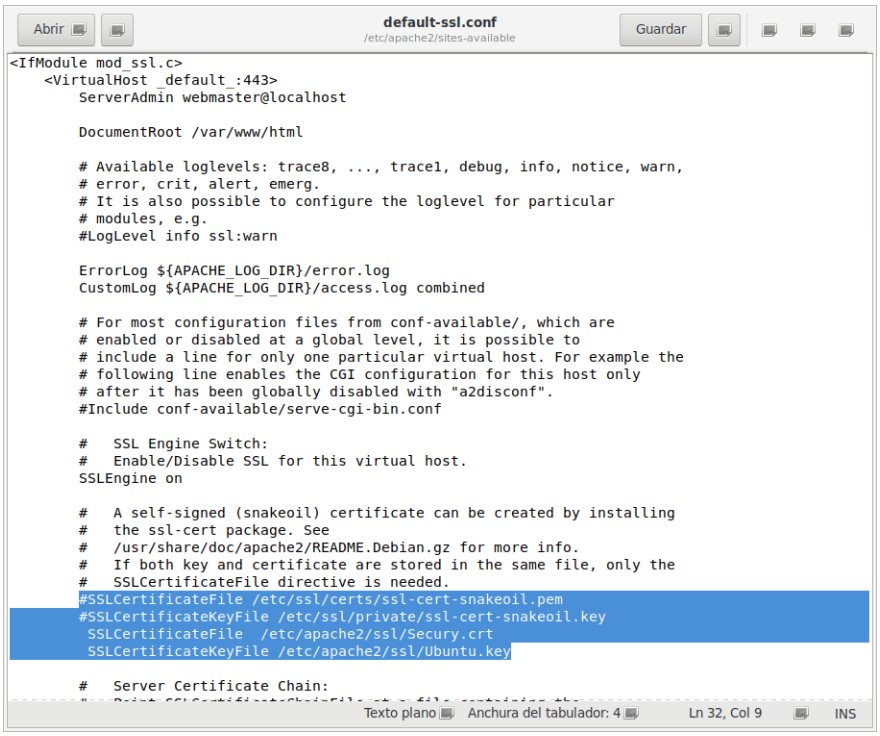

Figure 9: default-ssl.conf after modifications

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
```

Figure 10: As we can see, Listen 403 is uncommented, so it will works

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:

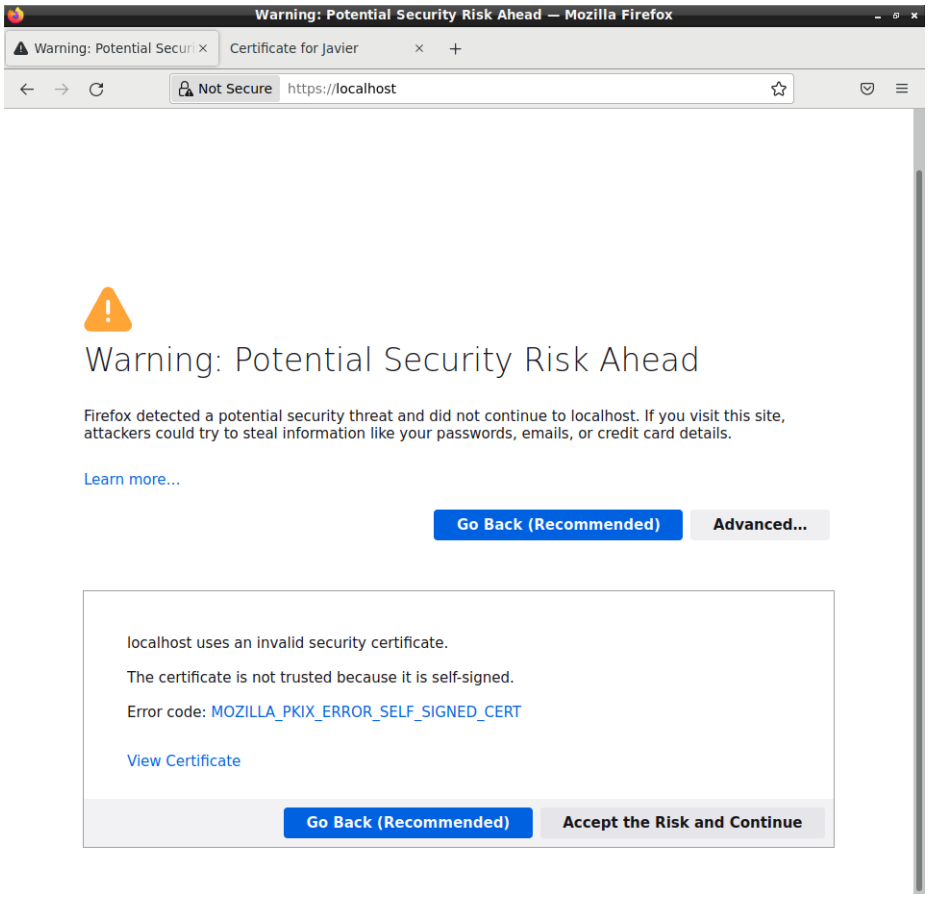

Figure 11: It says that it is not secure, let's see in *View Certificate*

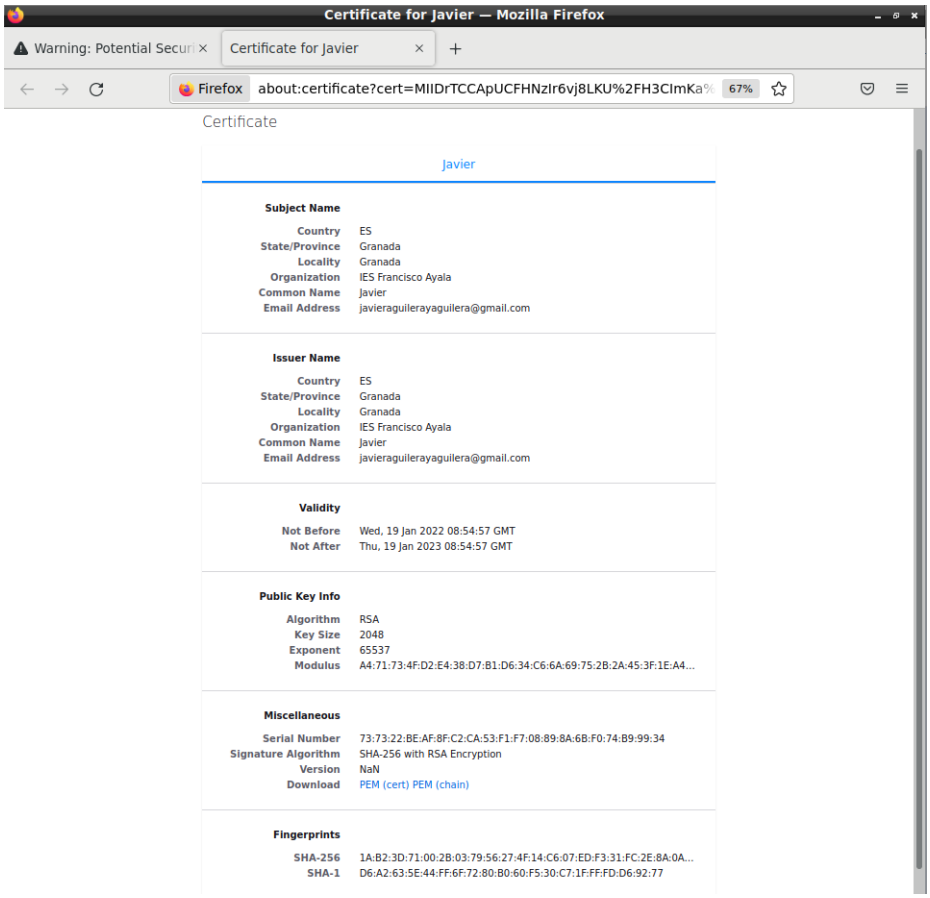

Figure 12: Our certificate information

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.

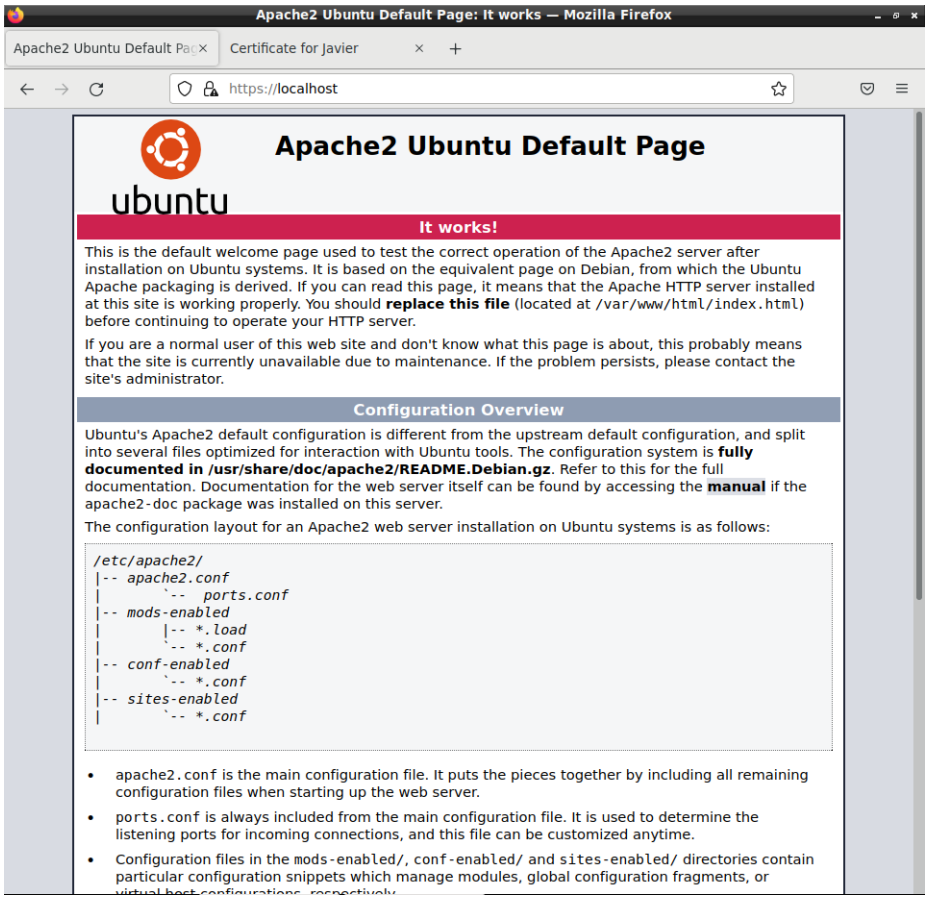

Figure 13: It works!

References

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