

**\*\*IMPORTANTE LEER POR FAVOR\*\***

Para hacer este ejercicio había que crear una cuenta en AWS y para poder crearla tienes que poner tu tarjeta bancaria, entonces no lo he podido hacer así. Sin embargo, he realizado el ejercicio localmente desde mi máquina de Ubuntu para cumplir con el objetivo de este ejercicio que es configurar Apache con SSL, crear certificados SSL y aprender el uso de dominios y DNS dinámico.

---

**1- Sigue las instrucciones: <https://josejuansanchez.org/iaw/practica-01-04/index.html>**

**Create un certificado ssl. Activa el módulo ssl, configura el virtual host ssl y comprueba que todo funciona correctamente.**

Actualizamos servidor:

```
fran@FranDesplie:~$ sudo apt update && sudo apt upgrade -y
[sudo] password for fran:
Hit:1 http://es.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://es.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://es.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
fran@FranDesplie:~$
```

Activamos el modulo SSL:

```
fran@FranDesplie:~$ sudo a2enmod ssl
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
fran@FranDesplie:~$
```

Luego, reiniciamos:

```
fran@FranDesplie:~$ sudo systemctl restart apache2
fran@FranDesplie:~$ _
```

Creamos una carpeta para los certificados:

```
fran@FranDesplie:~$ sudo mkdir /etc/apache2/ssl
fran@FranDesplie:~$
```

Ahora, generamos la clave privada y el certificado autofirmado completando la información que nos va pidiendo:

[illegible]

Configuramos el virtual host:

```
franc@FranDesplie:~$ sudo cp /etc/apache2/sites-available/default-ssl.conf /etc/apache2/sites-available/web-ssl.conf
franc@FranDesplie:~$ _
```

```

GNU nano 7.2 /etc/apache2/sites-available/web-ssl.conf *
<VirtualHost *:443>
    ServerAdmin fran@despliegue.local
    DocumentRoot /var/www/html
    ServerName localhost

    # 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/apache2/ssl/fran_certificado.crt
    SSLCertificateKeyFile /etc/apache2/ssl/fran_certificado.key

```

Habilitamos la web y reiniciamos:

```

fran@FranDesplie:~$ sudo a2ensite web-ssl.conf
Enabling site web-ssl.
To activate the new configuration, you need to run:
  systemctl reload apache2
fran@FranDesplie:~$ sudo systemctl restart apache2
fran@FranDesplie:~$

```

Ahora, comprobamos con `curl -v localhost` que nos muestra que el procedimiento del SSL:

```

fran@FranDesplie:~$ curl -v https://localhost
* Host localhost:443 was resolved.
* IPo6: ::1
* IPo4: 127.0.0.1
* Trying [::1]:443...
* Connected to localhost (::1) port 443
* ALPN: curl offers h2,http/1.1
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* CAfile: /etc/ssl/certs/ca-certificates.crt
* Capath: /etc/ssl/certs
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8):
* TLSv1.3 (IN), TLS handshake, Certificate (11):
* TLSv1.3 (OUT), TLS alert, unknown CA (560):
* SSL certificate problem: self-signed certificate
* Closing connection
curl: (60) SSL certificate problem: self-signed certificate
More details here: https://curl.se/docs/sslcerts.html

curl failed to verify the legitimacy of the server and therefore could not
establish a secure connection to it. To learn more about this situation and
how to fix it, please visit the web page mentioned above.
fran@FranDesplie:~$ _

```

**2- Sigue las instrucciones de: <https://josejuansanchez.org/iaw/practica-https/index.html>**

**Regístrate en no-ip y configura los DNS para que apunten a la dirección pública de la instancia. A continuación instala certbot y comprueba que todo funciona correctamente. Documenta la práctica en Github.**

Para esta parte he seguido todos los pasos para poder realizarla correctamente como:

Crear un hostname en No-IP (fran16121993.ddns.net).

Instalar Certbot y configurar Apache.

Intentar generar un certificado Let's Encrypt con `sudo certbot --apache -d fran16121993.ddns.net` (mi no-ip).

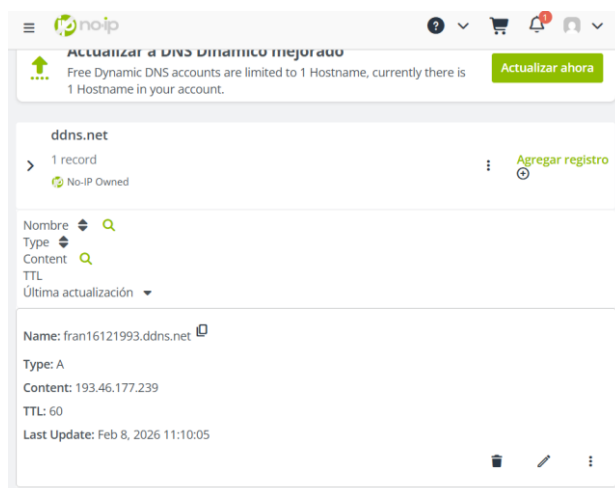
La validación no funcionó porque mi servidor está dentro de VirtualBox usando NAT, lo que significa que tiene una IP que solo funciona dentro de mi máquina. Por eso, Let's Encrypt no pudo conectarse desde internet para comprobar el dominio y emitir un certificado real.

En un entorno con servidor accesible públicamente el procedimiento funcionaría correctamente, y el certificado perfectamente.

Como comenté al principio de esta tarea, no pude hacer el ejercicio con AWS porque el crear la cuenta cloud te requiere una tarjeta bancaria. Entonces, me ha servido hacerlo así porque para ambas partes he aprendido los pasos necesarios para poder realizar correctamente.

Dejo las capturas de esta parte en local:

Me creé la cuenta en no-ip:



Verifico que el DNS apunte correctamente haciéndole un ping:

```
fran@FranDesplie:~$ ping fran16121993.ddns.net
PING fran16121993.ddns.net (193.46.177.239) 56(84) bytes of data.
64 bytes from 193.46.177.239: icmp_seq=1 ttl=255 time=2.77 ms
64 bytes from 193.46.177.239: icmp_seq=2 ttl=255 time=7.49 ms
64 bytes from 193.46.177.239: icmp_seq=3 ttl=255 time=1.76 ms
64 bytes from 193.46.177.239: icmp_seq=4 ttl=255 time=2.51 ms
64 bytes from 193.46.177.239: icmp_seq=5 ttl=255 time=2.39 ms
64 bytes from 193.46.177.239: icmp_seq=6 ttl=255 time=3.21 ms
64 bytes from 193.46.177.239: icmp_seq=7 ttl=255 time=2.04 ms
-
```

Instalo ahora el certbot y el plugin de apache:

```
fran@FranDesplie:~$ sudo apt install certbot python3-certbot-apache -y
[sudo] password for fran:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  Augeas-lenses libaugeas0 python3-acme python3-augeas python3-certbot python3-configargparse python3-icu python3-jinja2
Suggested packages:
  Augeas-doc python-certbot-doc python3-certbot-nginx Augeas-tools python-acme-doc python-certbot-apache-doc
The following NEW packages will be installed:
  Augeas-lenses certbot libaugeas0 python3-acme python3-augeas python3-certbot python3-certbot-apache python3-configargparse
python3-jinja2 python3-parsedatetime python3-rfc3339
0 upgraded, 12 newly installed, 0 to remove and 0 not upgraded.
Need to get 1657 kB of archives.
After this operation, 8599 kB of additional disk space will be used.
Get:1 http://es.archive.ubuntu.com/ubuntu/noble/universe amd64 Augeas-lenses all 1.14.1-1build2 [323 kB]
0% 11 Augeas-lenses 8348 B/323 kB 3%1_
```

Y como paso final, ejecutamos el certbot para generar el certificado:

```
fran@FranDesplie:~$ sudo certbot --apache -d fran16121993.ddns.net
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Enter email address (used for urgent renewal and security notices)
(Enter 'c' to cancel): fran@correo.local

-----
Please read the Terms of Service at
https://letsencrypt.org/documents/LE-SA-v1.6-August-18-2025.pdf. You must agree
in order to register with the ACME server. Do you agree?
-----
(Y)es/(N)o: y
There seem to be problems with that address. Enter email address (used for
urgent renewal and security notices)

If you really want to skip this, you can run the client with
--register-unsafely-without-email but you will then be unable to receive notice
about impending expiration or revocation of your certificates or problems with
your Certbot installation that will lead to failure to renew.

(Enter 'c' to cancel): lepero.fran@gmail.com

-----
Please read the Terms of Service at
https://letsencrypt.org/documents/LE-SA-v1.6-August-18-2025.pdf. You must agree
in order to register with the ACME server. Do you agree?
-----
(Y)es/(N)o: y

-----
Would you be willing, once your first certificate is successfully issued, to
share your email address with the Electronic Frontier Foundation, a founding
partner of the Let's Encrypt project and the non-profit organization that
develops Certbot? We'd like to send you email about our work encrypting the web,
EFF news, campaigns, and ways to support digital freedom.
-----
(Y)es/(N)o: n
Account registered.
Requesting a certificate for fran16121993.ddns.net
```

Y después de esto, no puedo comprobar el certificado porque el servidor no es accesible desde Internet. Entonces, con AWS, funcionaría fácil al sí ser accesible

desde internet, por lo demás, la idea y los pasos que he realizado reflejan como se haría por lo que he podido aprender como es el funcionamiento.