Práctica P02-PKI para HTTPS

Gabriel Lazovsky

Alejandro Rodríguez Ferrer

CEU

Índice

Parte obligatoria:

1-Comprobar que la máquina virtual Linux cuenta con fecha y hora válidas. Si no es así, actualizar la fecha/hora del sistema. Puede utilizarse el comando date –s, o NTP.

Máquina Rocky:

- Comprobamos la hora de la máquina:

```
[gabriel@localhost ~]$ date
vie 17 oct 2025 18:42:24 CEST
```

a) Instalamos chrony -> Para actualizar la localización de la máquina:

```
[gabriel@localhost ~]$ sudo apt update
[gabriel@localhost ~]$ sudo yum install chrony -y
[gabriel@localhost ~]$ sudo systemctl enable chronyd
[gabriel@localhost ~]$ sudo systemctl start chronyd
[gabriel@localhost ~]$ timedatectl
               Local time: vie 2025-10-17 18:45:49 CEST
           Universal time: vie 2025-10-17 16:45:49 UTC
                 RTC time: dom 2025-10-19 12:44:31
                Time zone: Europe/Madrid (CEST, +0200)
System clock synchronized: no
              NTP service: active
          RTC in local TZ: no
[gabriel@localhost ~]$ chronyc tracking
Reference ID    : 05FABFAA (ntp1.adora.codes)
Stratum         : 3
Ref time (UTC)  : Sun Oct 19 08:43:07 2025
System time     : 144216.109375000 seconds slow of NTP time
Last offset     : -0.000068158 seconds
RMS offset      : 21398.953125000 seconds
Frequency       : 1.467 ppm fast
Residual freq   : +0.013 ppm
Skew            : 1.446 ppm
Root delay      : 0.029216543 seconds
Root dispersion : 0.003417481 seconds
Update interval : 64.2 seconds
Leap status     : Normal
```

- Actualizamos la hora de forma manual:

```
[root@localhost ~]# date -s "19 OCT 2025 15:42:00"
dom 19 oct 2025 15:42:00 CEST
```

Máquina Ubuntu:

- Comprobamos la hora:

```
root@ubuntup2:~# date -s "19 OCT 2025 15:44:00"
dom 19 oct 2025 15:44:00 UTC
```

- Instalamos el paquete ntp:

```
root@ubuntup2:~# sudo apt update
```

b) Sincronizamos:

```
root@ubuntup2:~# sudo timedatectl set-ntp true
root@ubuntup2:~# timedatectl status
               Local time: dom 2025-10-19 13:46:12 UTC
           Universal time: dom 2025-10-19 13:46:12 UTC
                 RTC time: dom 2025-10-19 13:46:13
                Time zone: Etc/UTC (UTC, +0000)
System clock synchronized: yes
              NTP service: active
          RTC in local TZ: no
root@ubuntup2:~# sudo timedatectl set-timezone Europe/Madrid
root@ubuntup2:~# timedatectl status
               Local time: dom 2025-10-19 15:46:51 CEST
           Universal time: dom 2025-10-19 13:46:51 UTC
                 RTC time: dom 2025-10-19 13:46:52
                Time zone: Europe/Madrid (CEST, +0200)
System clock synchronized: yes
              NTP service: active
          RTC in local TZ: no
root@ubuntup2:~# date
dom 19 oct 2025 15:46:58 CEST
```

2. Instalar en la VM el servidor Apache y activar el módulo SSL. Iniciar el servidor y comprobar que acepta conexiones HTTPS. Describir las características del certificado que envía el servidor. Justificar las razones de la alerta que nos muestra nuestro navegador.

Ahora instalaremos Apache y activamos el módulo SSL en ambas máquinas:

- Rocky:

yum install httpd -y
a2enmod ssl

```
[root@localhost ~]# sudo yum install httpd -y
```

```
[root@localhost ~]# yum install mod_ssl
Última comprobación de caducidad de metadatos hecha hace 0:08:38, el mié 29 oct 2025 13:18:36.
Dependencias resueltas.
================================================================================
 Paquete          Arquitectura      Versión                    Repositorio      Tam.
================================================================================
Instalando:
 mod_ssl          aarch64           1:2.4.63-1.el10_0.2        appstream       102 k
Instalando dependencias:
 sscg             aarch64           3.0.5-9.el10               appstream        45 k

Resumen de la transacción
================================================================================
Instalar  2 Paquetes

Tamaño total de la descarga: 147 k
Tamaño instalado: 386 k
¿Está de acuerdo [s/N]?: s
Descargando paquetes:
(1/2): sscg-3.0.5-9.el10.aarch64.rpm                245 kB/s |  45 kB     00:00
(2/2): mod_ssl-2.4.63-1.el10_0.2.aarch64.rpm        536 kB/s | 102 kB     00:00
--------------------------------------------------------------------------------
Total                                               368 kB/s | 147 kB     00:00
Ejecutando verificación de operación
Verificación de operación exitosa.
```

- Reiniciamos y habilitamos http:

```
[root@localhost ~]# systemctl start httpd
[root@localhost ~]# systemctl status httpd
● httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: d
isabled)
     Active: active (running) since Sun 2025-10-19 15:53:29 CEST; 5min ago
 Invocation: 9f55793ac6ce4a369069697f1c5bff48
       Docs: man:httpd.service(8)
   Main PID: 5664 (httpd)
     Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Byte
s served/sec:   >
      Tasks: 177 (limit: 10685)
     Memory: 13.7M (peak: 14.4M)
        CPU: 640ms
     CGroup: /system.slice/httpd.service
             ├─5664 /usr/sbin/httpd -DFOREGROUND
             ├─5665 /usr/sbin/httpd -DFOREGROUND
             ├─5666 /usr/sbin/httpd -DFOREGROUND
             ├─5667 /usr/sbin/httpd -DFOREGROUND
             └─5668 /usr/sbin/httpd -DFOREGROUND
```

- Debemos habilitarlo por el firewall:

firewall-cmd --permanent --add-service=http
sudo firewall-cmd --permanent --add-service=https
sudo firewall-cmd --reload

```
[root@localhost ~]# sudo firewall-cmd --permanent --add-service=http
sudo firewall-cmd --permanent --add-service=https
sudo firewall-cmd --reload
success
success
success
```

```
[root@localhost ~]# sudo firewall-cmd --list-all
public (default, active)
  target: default
  ingress-priority: 0
  egress-priority: 0
  icmp-block-inversion: no
  interfaces: ens160
  sources:
  services: cockpit dhcpv6-client http https ssh
  ports:
  protocols:
  forward: yes
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

c) Ubuntu:

```
sudo apt install apache2 -y
```

- Reiniciamos y dejamos como enable

```
root@ubuntup2:~# systemctl restart apache2
root@ubuntup2:~# systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /usr/lib/
systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable apache2
root@ubuntup2:~# systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled>
     Active: active (running) since Sun 2025-10-19 15:53:04 CEST; 12m>
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 3543 (apache2)
      Tasks: 55 (limit: 4547)
     Memory: 5.9M (peak: 6.2M)
        CPU: 104ms
     CGroup: /system.slice/apache2.service
             ├─3543 /usr/sbin/apache2 -k start
             ├─3545 /usr/sbin/apache2 -k start
             └─3546 /usr/sbin/apache2 -k start

oct 19 15:53:04 ubuntup2 systemd[1]: Starting apache2.service - The A>
oct 19 15:53:04 ubuntup2 apachectl[3542]: AH00558: apache2: Could not>
oct 19 15:53:04 ubuntup2 systemd[1]: Started apache2.service - The Ap>
```

- Para comprobar que hemos habilitado el puerto 443:

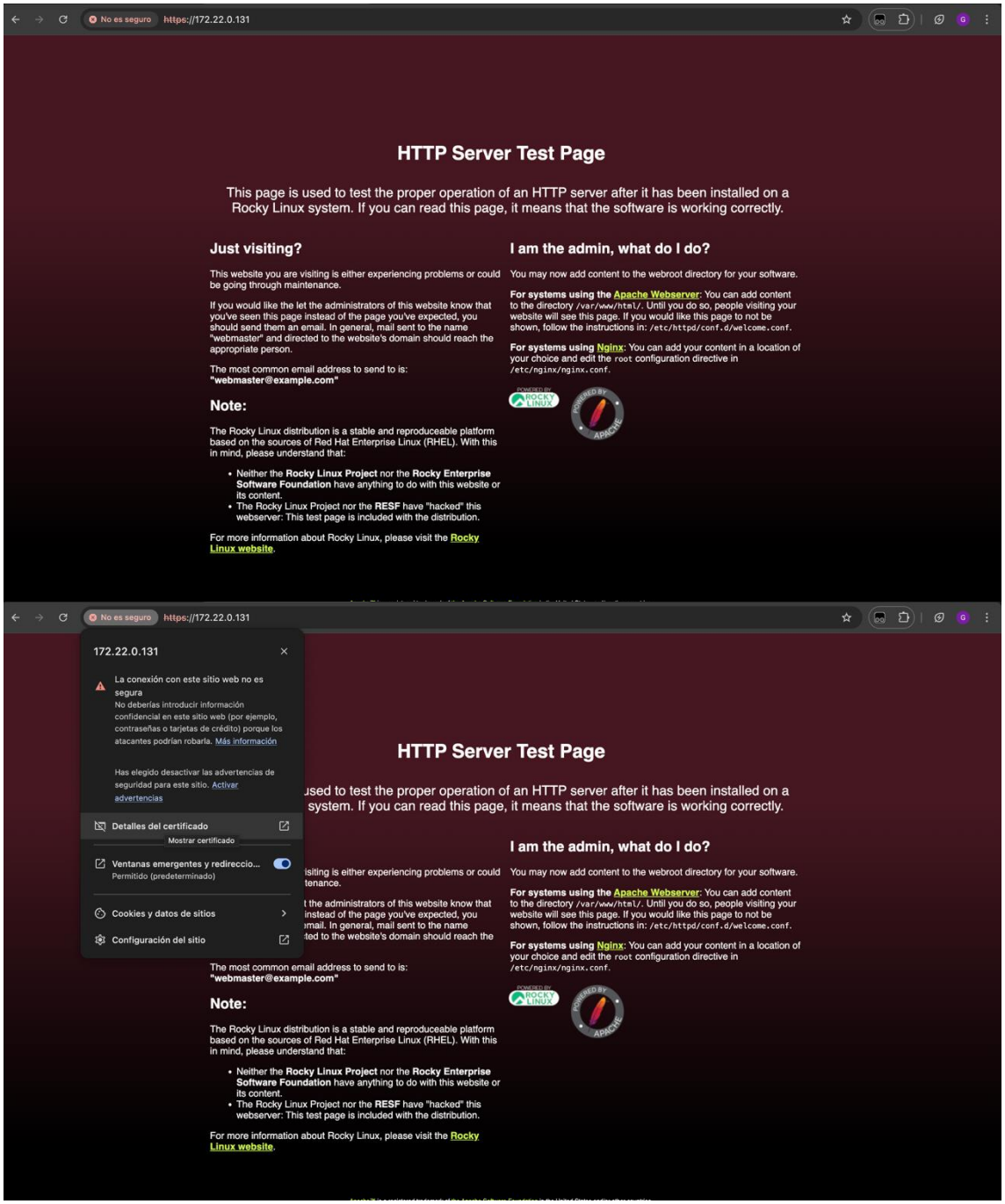
```
[root@localhost ~]# netstat -unltp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
  PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
  944/sshd: /usr/sbin
tcp6       0      0 :::80                   :::*                    LISTEN
  5664/httpd
tcp6       0      0 :::22                   :::*                    LISTEN
  944/sshd: /usr/sbin
tcp6       0      0 :::443                  :::*                    LISTEN
  5664/httpd
udp        0      0 127.0.0.1:323           0.0.0.0:*
  866/chronyd
● httpd.service - The Apache HTTP Server
```

root@ubuntup2:~# sudo a2enmod ssl
sudo a2ensite default-ssl
sudo systemctl restart apache2
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Module socache_shmcb already enabled
Module ssl already enabled
Enabling site default-ssl.
To activate the new configuration, you need to run:
  systemctl reload apache2
root@ubuntup2:~# systemctl reload apache2

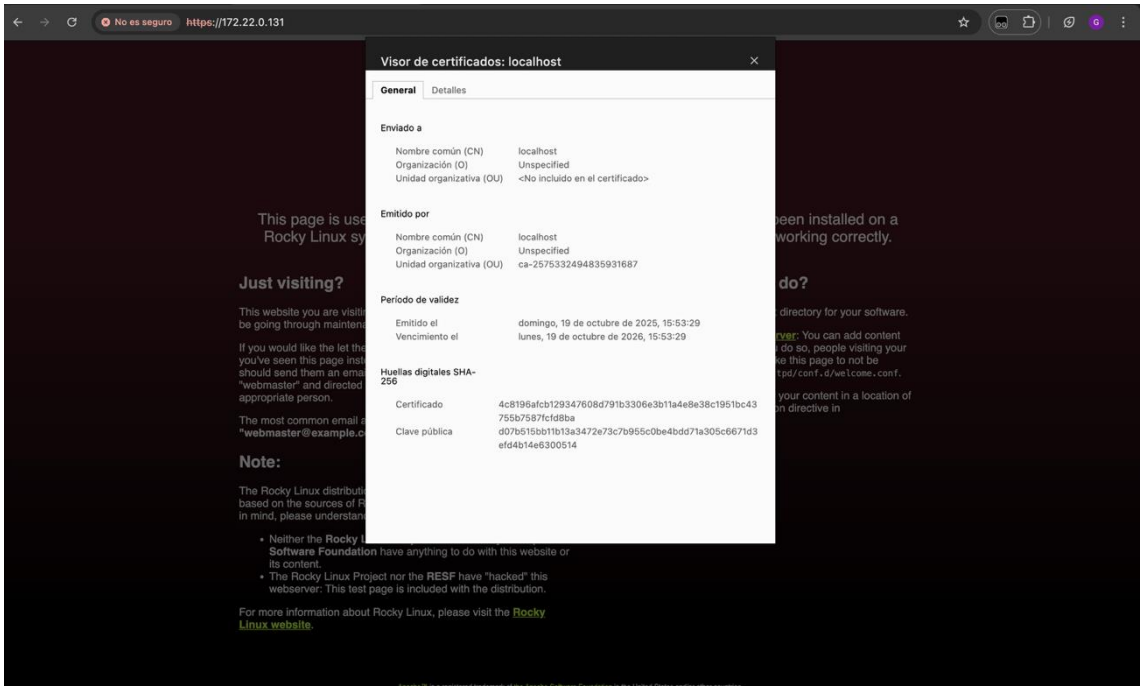
```
root@ubuntup2:~# sudo firewall-cmd --permanent --add-service=http
sudo firewall-cmd --permanent --add-service=https
sudo firewall-cmd --reload
sudo: firewall-cmd: command not found
sudo: firewall-cmd: command not found
sudo: firewall-cmd: command not found
root@ubuntup2:~# sudo a2enmod ssl
sudo a2ensite default-ssl
sudo systemctl restart apache2
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Module socache_shmcb already enabled
Module ssl already enabled
Enabling site default-ssl.
To activate the new configuration, you need to run:
  systemctl reload apache2
root@ubuntup2:~# systemctl reload apache2
root@ubuntup2:~# locate openssl.cnf
Command 'locate' not found, but can be installed with:
apt install plocate
root@ubuntup2:~# apt install plocate
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  liburing2
```

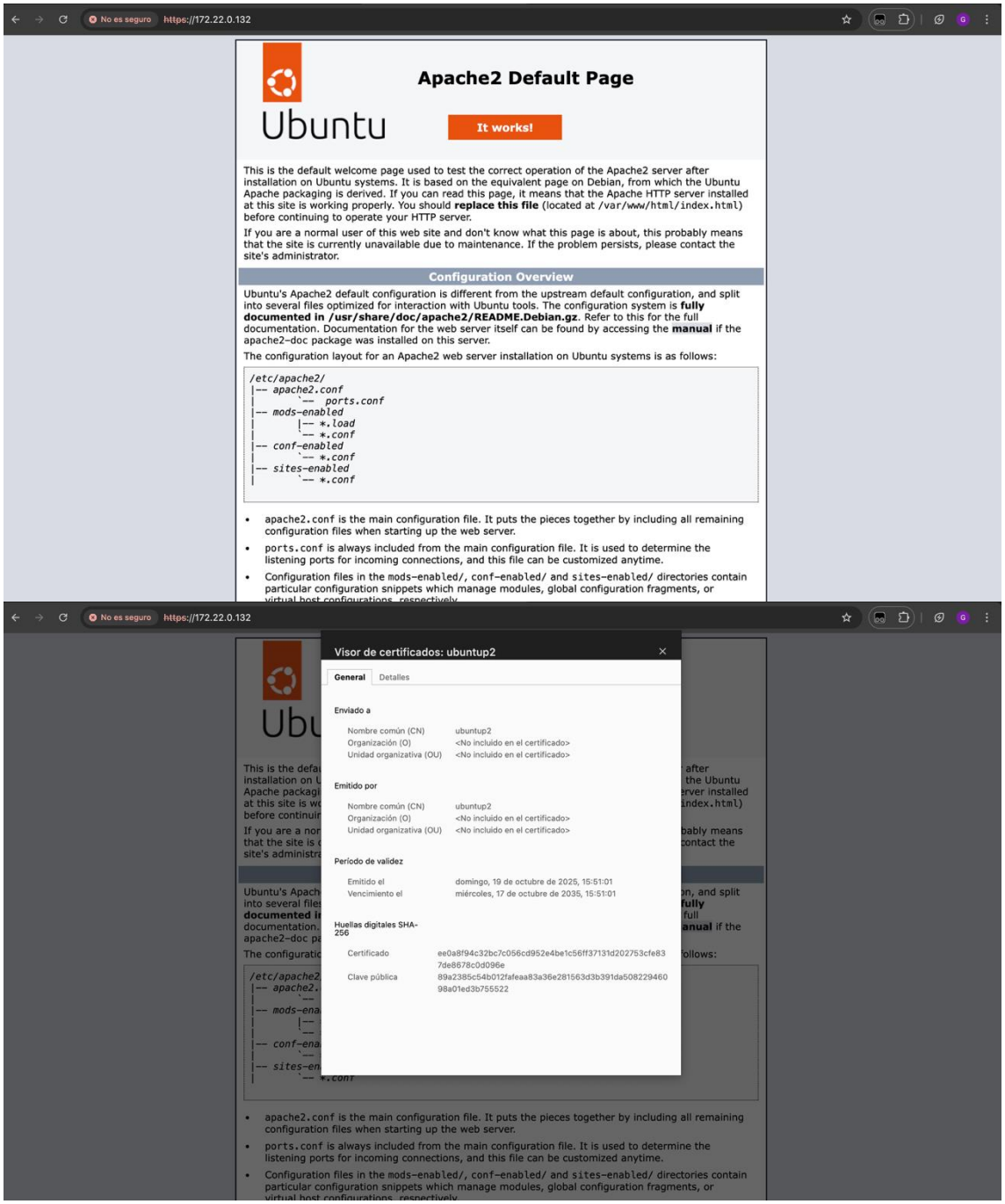
```
# pkix   : PrintableString, BMPString (PKIX recommendation before 2004)
# utf8only: only UTF8Strings (PKIX recommendation after 2004).
# nombstr : PrintableString, T61String (no BMPStrings or UTF8Strings).
# MASK:XXXX a literal mask value.
# WARNING: ancient versions of Netscape crash on BMPStrings or UTF8Strings.
string_mask = utf8only

# req_extensions = v3_req # The extensions to add to a certificate request

[ req_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_default             = AU
countryName_min                 = 2
countryName_max                 = 2

stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default     = Some-State

localityName                    = Locality Name (eg, city)

0.organizationName              = Organization Name (eg, company)
0.organizationName_default      = Internet Widgits Pty Ltd

# we can do this but it is not needed normally :-)
#1.organizationName             = Second Organization Name (eg, company)
#1.organizationName_default     = World Wide Web Pty Ltd
                                                              173,40-44     42%

# activate = 1


####################################################################
[ ca ]
default_ca      = CA_default            # The default ca section

####################################################################
[ CA_default ]

dir             = ./demoCA              # Where everything is kept
certs           = $dir/certs            # Where the issued certs are kept
crl_dir         = $dir/crl              # Where the issued crl are kept
database        = $dir/index.txt        # database index file.
#unique_subject = no                    # Set to 'no' to allow creation of
                                        # several certs with same subject.
new_certs_dir   = $dir/newcerts         # default place for new certs.

certificate     = $dir/cacert.pem       # The CA certificate
serial          = $dir/serial           # The current serial number
crlnumber       = $dir/crlnumber        # the current crl number
                                        # must be commented out to leave a V1 CRL
crl             = $dir/crl.pem          # The current CRL
private_key     = $dir/private/cakey.pem# The private key

x509_extensions = usr_cert              # The extensions to add to the cert
                                                              81,0-1        19%
```

```
root@ubuntup2:~# apt install net-tools
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes NUEVOS:
  net-tools
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 65 no actualizados.
Se necesita descargar 207 kB de archivos.
Se utilizarán 1.229 kB de espacio de disco adicional después de esta operación
.
Des:1 http://ports.ubuntu.com/ubuntu-ports noble-updates/main arm64 net-tools
arm64 2.10-0.1ubusudo a2enmod ssl
sudo a2ensite default-sslewall-cmd --permanent --add-service=http
Scanning processes... [
root@ubuntup2:~# locate openssl.cnfypervisor (qemu) binaries on this host.
/etc/ssl/openssl.cnf
/usr/lib/shim/mok/openssl.cnf
/usr/lib/ssl/openssl.cnf
```


No es seguro
https://172.22.0.131
HTTP Server Test Page
This page is used to test the proper operation of an HTTP server after it has been installed on a Rocky Linux system. If you can read this page, it means that the software is working correctly.
Just visiting?
This website you are visiting is either experiencing problems or could be going through maintenance.
If you would like the let the administrators of this website know that you've seen this page instead of the page you've expected, you should send them an email. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.
The most common email address to send to is: "webmaster@example.com"
Note:
The Rocky Linux distribution is a stable and reproduceable platform based on the sources of Red Hat Enterprise Linux (RHEL). With this in mind, please understand that:
Neither the Rocky Linux Project nor the Rocky Enterprise Software Foundation have anything to do with this website or its content.
The Rocky Linux Project nor the RESF have "hacked" this webserver: This test page is included with the distribution.
For more information about Rocky Linux, please visit the Rocky Linux website.
I am the admin, what do I do?
You may now add content to the webroot directory for your software.
For systems using the Apache Webserver: You can add content to the directory /var/www/html/. Until you do so, people visiting your website will see this page. If you would like this page to not be shown, follow the instructions in: /etc/httpd/conf.d/welcome.conf.
For systems using Nginx: You can add your content in a location of your choice and edit the root configuration directive in /etc/nginx/nginx.conf.
POWERED BY ROCKY LINUX
POWERED BY APACHE

No es seguro
https://172.22.0.131
172.22.0.131
La conexión con este sitio web no es segura
No deberías introducir información confidencial en este sitio web (por ejemplo, contraseñas o tarjetas de crédito) porque los atacantes podrían robarla. Más información
Has elegido desactivar las advertencias de seguridad para este sitio. Activar advertencias
Detalles del certificado
Mostrar certificado
Ventanas emergentes y redireccio...
Permitido (predeterminado)
Cookies y datos de sitios
Configuración del sitio
HTTP Server Test Page
system. If you can read this page, it means that the software is working correctly.
I am the admin, what do I do?
You may now add content to the webroot directory for your software.
For systems using the Apache Webserver: You can add content to the directory /var/www/html/. Until you do so, people visiting your website will see this page. If you would like this page to not be shown, follow the instructions in: /etc/httpd/conf.d/welcome.conf.
For systems using Nginx: You can add your content in a location of your choice and edit the root configuration directive in /etc/nginx/nginx.conf.
The most common email address to send to is: "webmaster@example.com"
Note:
The Rocky Linux distribution is a stable and reproduceable platform based on the sources of Red Hat Enterprise Linux (RHEL). With this in mind, please understand that:
Neither the Rocky Linux Project nor the Rocky Enterprise Software Foundation have anything to do with this website or its content.
The Rocky Linux Project nor the RESF have "hacked" this webserver: This test page is included with the distribution.
For more information about Rocky Linux, please visit the Rocky Linux website.


No es seguro
https://172.22.0.131
Visor de certificados: localhost
General
Detalles
Enviado a
Nombre común (CN) localhost
Organización (O) Unspecified
Unidad organizativa (OU) <No incluido en el certificado>
Emitido por
Nombre común (CN) localhost
Organización (O) Unspecified
Unidad organizativa (OU) ca-2575332494835931687
Período de validez
Emitido el domingo, 19 de octubre de 2025, 15:53:29
Vencimiento el lunes, 19 de octubre de 2026, 15:53:29
Huellas digitales SHA-256
Certificado 4c8196afcb129347608d791b3306e3b11a4e8e38c1951bc43755b7587fcfd8ba
Clave pública d07b515bb11b13a3472e73c7b955c0be4bdd71a305c6671d3efd4b14e6300514
Just visiting?
Note:
Rocky Linux website


No es seguro
https://172.22.0.132
Apache2 Default Page
Ubuntu
It works!
This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should replace this file (located at /var/www/html/index.html) before continuing to operate your HTTP server.
If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.
Configuration Overview
Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is fully documented in /usr/share/doc/apache2/README.Debian.gz. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the manual if the apache2-doc package was installed on this server.
The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:
/etc/apache2/
|-- apache2.conf
| `-- ports.conf
|-- mods-enabled
| |-- *.load
| `-- *.conf
|-- conf-enabled
| `-- *.conf
|-- sites-enabled
| `-- *.conf
apache2.conf is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
ports.conf is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
Configuration files in the mods-enabled/, conf-enabled/ and sites-enabled/ directories contain particular configuration snippets which manage modules, global configuration fragments, or
No es seguro
https://172.22.0.132
Visor de certificados: ubuntup2
General
Detalles
Enviado a
Nombre común (CN) ubuntup2
Organización (O) <No incluido en el certificado>
Unidad organizativa (OU) <No incluido en el certificado>
Emitido por
Nombre común (CN) ubuntup2
Organización (O) <No incluido en el certificado>
Unidad organizativa (OU) <No incluido en el certificado>
Período de validez
Emitido el domingo, 19 de octubre de 2025, 15:51:01
Vencimiento el miércoles, 17 de octubre de 2035, 15:51:01
Huellas digitales SHA-256
Certificado ee0a8f94c32bc7c056cd952e4be1c56ff37131d202753cfe837de8678c0d096e
Clave pública 89a2385c54b012fafeaa83a36e281563d3b391da50822946098a01ed3b755522

3. Identificar la ubicación del archivo openssl.cnf. Crear una Autoridad de Certificación propia, con los siguientes parámetros por defecto: CountryName (ES), StateOrProvinceName (Madrid), LocalityName (Alcorcon) y OrganizationName (CA de Pruebas). A continuación, obtendremos las claves pública y privada de la CA (archivo cakey.pem, protegido por frase de paso) y un certificado autofirmado con CN = “CA del CEU para Pruebas” (cacert.pem). Por último, obtenemos el listado de certificados revocados (crl.pem) que inicialmente estará vacío.

-Ubuntu

```
root@ubuntup2:~#  mkdir -p /etc/pki/tls
root@ubuntup2:~# cd /etc/pki/tls
```

```
root@ubuntup2:/etc/pki/tls# mkdir certs
root@ubuntup2:/etc/pki/tls# mkdir crl
root@ubuntup2:/etc/pki/tls# mkdir newcerts
root@ubuntup2:/etc/pki/tls# mkdir private
root@ubuntup2:/etc/pki/tls# touch index.txt
root@ubuntup2:/etc/pki/tls# echo 01 > serial
echo 01 > crlnumber
root@ubuntup2:/etc/pki/tls# ls
certs  crl  crlnumber  index.txt  newcerts  private  serial
```

openssl genrsa -aes256 -out private/cakey.pem 2048
openssl req -new -key private/cakey.pem -out ca-csr.pem
openssl req -x509 -extensions v3_ca -in ca-csr.pem -out cacert.pem -key private/cakey.pem -days 3652

Y el listado de revocados:

openssl ca -gencrl -out crl.pem

-Rocky:

dnf install openssl -y

Comenzamos creando creando las carpetas: certs,crl,newcerts,private y dándoles permisos.

cd /etc/pki/tls/
mkdir certs crl newcerts private
touch index.txt
echo 01 > serial
echo 01 > crinumber

```
[root@localhost tls]# ls -l
total 40
drwxr-xr-x. 2 root root  8192 oct 19 15:53 certs
-rw-r--r--. 1 root root     3 oct 19 16:46 crinumber
drwxr-xr-x. 2 root root     6 oct 19 16:44 crl
-rw-r--r--. 1 root root   412 ene 29  2025 ct_log_list.cnf
lrwxrwxrwx. 1 root root    50 ene 29  2025 fips_local.cnf -> /etc/crypto-policies/back-ends/openssl_fips.config
-rw-r--r--. 1 root root     0 oct 19 16:45 index.txt
drwxr-xr-x. 2 root root     6 ene 29  2025 misc
drwxr-xr-x. 2 root root     6 oct 19 16:44 newcerts
-rw-r--r--. 1 root root 12414 oct 19 16:53 openssl.cnf
drwxr-xr-x. 2 root root     6 ene 29  2025 openssl.d
drwxr-xr-x. 2 root root    27 oct 19 15:53 private
-rw-r--r--. 1 root root     3 oct 19 16:45 serial
```

- Dentro de openssl.conf, sustituimos los default por los valores del enunciado:

```
[ req_distinguished_name ]
countryName                     = Cour
 code)
countryName_default             = ES
stateOrProvinceName_default     = Madrid
localityName_default            = Alcorcon
0.organizationName_default      = CA de Pruebas
```

- Dentro de /etc/httpd/conf.d/ssl.conf , modificaremos estas rutas:

```
SSLCertificateKeyFile /etc/pki/tls/prueba-key.pem
```

-

```
SSLCertificateFile /etc/pki/tls/prueba-crt.pem
```

-

```
SSLCertificateFile /etc/pki/tls/prueba-crt.pem

#   Server Private Key:
#   If the key is not combined with the certificate, use this
#   directive to point at the key file.  Keep in mind that if
#   you've both a RSA and a DSA private key you can configure
#   both in parallel (to also allow the use of DSA ciphers, etc.)
#   ECC keys, when in use, can also be configured in parallel
SSLCertificateKeyFile /etc/pki/tls/prueba-key.pem
```

openssl genrsa -aes256 -out private/cakey.pem 2048
openssl req -new -key private/cakey.pem -out ca-csr.pem
openssl req -x509 -extensions v3_ca -in ca-csr.pem -out cacert.pem -key private/cakey.pem -days 3652

Y el listado de revocados:

openssl ca -gencrl -out crl.pem

Copia de seguridad (Opcional):

sudo cp /etc/pki/tls/openssl.cnf /etc/pki/tls/openssl.cnf.orig

Modificamos openssl.cnf de la siguiente manera:

```
[ CA_default ]

dir          = /etc/pki/tls        # ESTE LO MODIFICAMOS A LA RUTA QUE HAYAMOS
CREADO
certs        = $dir/certs          # Where the issued certs are kept
crl_dir      = $dir/crl            # Where the issued crl are kept
database     = $dir/index.txt      # database index file.
#unique_subject = no                # Set to 'no' to allow creation of
                                    # several certs with same subject.
new_certs_dir   = $dir/newcerts        # default place for new certs.

certificate    = $dir/cacert.pem      # The CA certificate
serial       = $dir/serial         # The current serial number
crlnumber     = $dir/crlnumber       # the current crl number
                                    # must be commented out to leave a V1 CRL
crl          = $dir/crl.pem         # The current CRL
private_key    = $dir/private/cakey.pem # The private key

x509_extensions = usr_cert

[ req ]
subjectAltName = @alt_names
default_bits         = 2048
default_md           = sha256
default_keyfile      = privkey.pem
distinguished_name     = req_distinguished_name
attributes           = req_attributes
x509_extensions = v3_ca # The extensions to add to the self signed cert
req_extensions = v3_req


[ req_distinguished_name ]
countryName                = Country Name (2 letter code)
countryName_default          = ES


stateOrProvinceName          = State or Province Name (full name)
stateOrProvinceName_default    = Madrid

localityName               = Locality Name (eg, city)
localityName_default         = Alcorcon

0.organizationName           = Organization Name (eg, company)
```

```
0.organizationName_default     = CA de Pruebas

[ usr_cert ]

basicConstraints=CA:FALSE
extendedKeyUsage = clientAuth
keyUsage = digitalSignature, keyEncipherment
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer

[ v3_req ]
basicConstraints = CA:FALSE
keyUsage = digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth
subjectAltName = @alt_names

[ v3_ca ]

basicConstraints = critical, CA:TRUE, pathlen:3
# Identificadores clave/autoridad
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
# Uso de clave: sólo firma de certificados y CRLs
keyUsage = critical, cRLSign, keyCertSign
# Opcional para interoperabilidad histórica
nsCertType = sslCA, emailCA

[ server_SAN ]
basicConstraints = CA:FALSE
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer:always
# Uso de la clave del servidor
keyUsage = digitalSignature, keyEncipherment
# Propósito extendido: autenticación de servidor TLS
extendedKeyUsage = serverAuth
# nsCertType histórico (opcional)
nsCertType = server

# subjectAltName debe apuntar a una subsección de 'alt_names' o a un valor en
línea
# Ejemplo de referencia a una subsección:
subjectAltName = @alt_names


[ alt_names ]
DNS.1 = www.miservidor.es
DNS.2 = miservidor.es
```

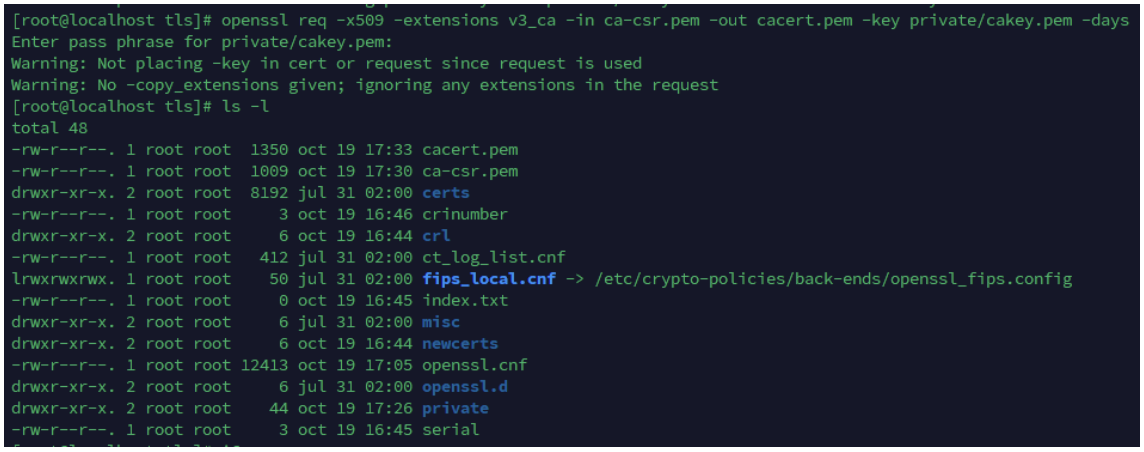
[crl_ext]
authorityKeyIdentifier=keyid:always

---

4. Crear una pareja de claves pública y privada RSA de 2048 bits para nuestro servidor web y un certificado firmado por nuestra CA. El CN (y el nombre alternativo SAN) deberá corresponder con el nombre del servidor (**www.miservidor.es**). Configurar Apache para que atienda las peticiones dirigidas al dominio por HTTPS (redirigir también las conexiones HTTP hacia HTTPS). La página de inicio estará en **/projects/miservidor**. Llevar a cabo los pasos necesarios para que nuestro cliente pueda conectar con el servidor vía HTTPS sin que se visualice ningún *warning* (reconociendo la CA y el certificado del servidor).

- Rocky/Ubuntu:

```
[root@localhost ~]# sudo openssl genrsa -aes256 -out /etc/pki/tls/private/cakey.pem
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
[root@localhost ~]# ls
anaconda-ks.cfg
[root@localhost ~]# cd /etc/pki/tls/
[root@localhost tls]# ls
certs       ct_log_list.cnf  misc         openssl.d
crinumber   fips_local.cnf   newcerts     private
crl         index.txt        openssl.cnf  serial
[root@localhost tls]# ls private/
cakey.pem  localhost.kopenssl req -new -key private/cakey.pem -out ca-csr.pem^C
sl.conf calhost tls]# openssl req -new -key private/cakey.pem -out ca-csr.pem
Enter pass phrase for private/cakey.pem:
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) [ES]:
State or Province Name (full name) [Madrid]:
Locality Name (eg, city) [Alcorcon]:
Organization Name (eg, company) [CA de Pruebas]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:CA del CEU para Pruebas
Email Address []:
```



Generar la clave privada del servidor y permisos para root:

```
sudo openssl genrsa -out /etc/pki/tls/private/miservidor.key 2048
sudo chmod 600 /etc/pki/tls/private/miservidor.key
```

Generar la solicitud de certificado (CSR) del servidor:

```
sudo openssl req -new -key /etc/pki/tls/private/miservidor.key \
  -out /etc/pki/tls/miservidor.csr
```

```
sudo mkdir -p /var/www/miservidor
echo "<h1>Servidor HTTPS activo con CA propia</h1>" | sudo tee
/var/www/miservidor/index.html
sudo chown -R apache:apache /var/www/miservidor
```

```
vim /etc/apache2/sites-available/miservidor.conf
```

```
<VirtualHost *:80>
   ServerName www.miservidor.es
   Redirect permanent / https://www.miservidor.es/
</VirtualHost>

<VirtualHost *:443>
   ServerName www.miservidor.es

   DocumentRoot /projects/miservidor

   SSLEngine on
   SSLCertificateFile      /etc/ssl/certs/miservidor.crt
   SSLCertificateKeyFile   /etc/ssl/private/miservidor.key
   SSLCertificateChainFile /etc/ssl/certs/ca.crt

   <Directory /projects/miservidor>
      Options Indexes FollowSymLinks
      AllowOverride All
      Require all granted
   </Directory>
</VirtualHost>
```

sudo a2ensite miservidor.conf
sudo systemctl reload apache2

Importamos los certificados a nuestra maquina anfitriona.

Ubuntu:

Rocky:

Generamos la clave privada RSA usando OpenSSL:

sudo openssl genrsa -aes256 -out /etc/pki/tls/private/cakey.pem 4096

Creamos un certificado raíz autofirmado CA:

```
sudo openssl req -config /etc/pki/tls/openssl.cnf \
    -new -x509 -days 3650 \
```

```
    -key /etc/pki/tls/private/cakey.pem \
    -out /etc/pki/tls/certs/cacert.pem \
   -extensions v3_ca
```

Valores:

```
Country Name (2 letter code) [ES]:
State or Province Name (full name) [Madrid]:
Locality Name (eg, city) [Alcorcon]:
Organization Name (eg, company) [CA de Pruebas]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname)
[]:www.miservidor.com
Email Address []:
```

Generar la clave privada del servidor y permisos para root:

```
sudo openssl genrsa -out /etc/pki/tls/private/miservidor.key 2048
sudo chmod 600 /etc/pki/tls/private/miservidor.key
```

Generar la solicitud de certificado (CSR) del servidor:

```
sudo openssl req -new -key /etc/pki/tls/private/miservidor.key \
  -out /etc/pki/tls/miservidor.csr
```

Revisar el CSR (opcional)

```
sudo openssl req -in /etc/pki/tls/miservidor.csr -noout -text | sed -n '1,120p'
```

Firmamos usando la CA:

```
sudo openssl ca -config /etc/pki/tls/openssl.cnf \
  -in /etc/pki/tls/miservidor.csr \
  -out /etc/pki/tls/certs/miservidor.crt \
  -days 825 -extensions server_SAN
```

Revisar CRT (opcional)

```
openssl x509 -in /etc/pki/tls/certs/miservidor.crt -noout -text | sed -n '1,220p'
```

Revisar BBDD V/R + Serial:

```
tail -n +1 /etc/pki/tls/index.txt
```

```
cat /etc/pki/tls/serial
```

Instalamos el paquete mod_ssl:

```
sudo dnf install -y mod_ssl
```

Arrancamos Apache:

```
sudo systemctl start httpd
sudo systemctl enable httpd
sudo systemctl restart httpd
```

Editamos/Creamos /etc/httpd/conf.d/miservidor.conf y añadimos:

```
<VirtualHost *:80>
    ServerName www.miservidor.es
    ServerAlias miservidor.es
    DocumentRoot /var/www/miservidor
    # Redirección automática a HTTPS
    RewriteEngine On
    RewriteCond %{HTTPS} off
    RewriteRule ^/?(.*) https://%{SERVER_NAME}/$1 [R=301,L]
</VirtualHost>

<VirtualHost *:443>
    ServerName www.miservidor.es
    ServerAlias miservidor.es
    DocumentRoot /var/www/miservidor

    SSLEngine on
    SSLCertificateFile /etc/pki/tls/certs/miservidor.crt
    SSLCertificateKeyFile /etc/pki/tls/private/miservidor.key
    SSLCACertificateFile /etc/pki/tls/certs/cacert.pem

    # === NUEVAS DIRECTIVAS PARA AUTENTICACIÓN CLIENTE ===
    SSLVerifyClient require
    SSLVerifyDepth 1
    SSLOptions +StdEnvVars
    # ================================================

    <Directory /var/www/miservidor>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>
</VirtualHost>
```

Creamos la carpeta /var/www/miservidor y añadimos lo siguiente de forma directa, más permisos:

```
sudo mkdir -p /var/www/miservidor
echo "<h1>Servidor HTTPS activo con CA propia</h1>" | sudo tee
/var/www/miservidor/index.html
sudo chown -R apache:apache /var/www/miservidor
```

Este comando lo usaremos para verificar la sintaxis en apache:

```
sudo apachectl configtest
```

"Could not reliably determine the server's fully qualified domain name, using ::1. Set the 'ServerName' directive globally to suppress this message
"

Este Warning se arregla agregando (ServerName www.miservidor.es) en /etc/httpd/conf/httpd.conf

Para aplicar la configuración reiniciamos el servicio:

```
sudo systemctl restart httpd
```

En el archivo en /etc/httpd/conf.d/ssl.conf modificaremos las siguientes rutas: ´

```
SSLCertificateFile /etc/pki/tls/certs/localhost.crt
SSLCertificateKeyFile /etc/pki/tls/private/localhost.key
```

```
sudo apachectl configtest
sudo systemctl restart httpd
```

En /etc/hosts añadimos nuestros dominios a 127.0.0.1:
server www.miservidor.es miservidor.es

En Windows:
172.22.0.134 www.miservidor.es miservidor.es

Podemos comprobar su funcionamiento usando curl o el browser:

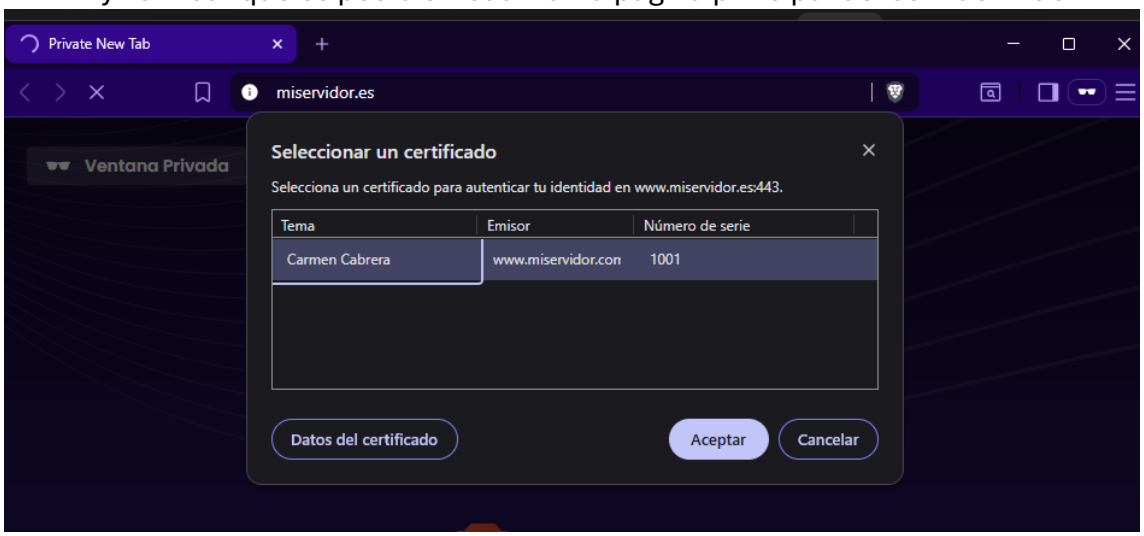
```
curl -vk --cacert /etc/pki/tls/certs/cacert.pem https://www.miservidor.es/
```

Ahora añadiremos el certificado a Windows:

1) sudo openssl x509 -in /etc/pki/tls/certs/cacert.pem -out /root/cacert.cer -outform DER
2) Desde PowerShell: scp root@172.22.134:/root/cacert.cer "C:\Users\Usuario\Downloads"
3) Tecla Windows + R y buscamos "certmgr.msc"
4) Carpeta Entidades de certificación raíz de confianza -> Acción -> Todas las tareas -> Importar

Ahora desde el Browser en modo incógnito deberíamos de conectarnos y el certificado estará activo.

5. Crear un certificado de cliente a nombre de Carmen Cabrera (CN), del departamento de Contabilidad (OU), firmado por nuestra CA. A continuación, configurar el servidor Apache para que requiera certificados cliente en la raíz del proyecto a todas las conexiones HTTPS. Establecer una conexión HTTPS con el navegador y verificar que se requiere el certificado de cliente para el acceso. Instalar el certificado en el navegador del cliente y verificar que es posible visualizar la página principal del servidor web.



Ubuntu:

openssl genrsa -out carmen.key 2048
chmod 600 carmen.key
Creamos un nuevo certificado:
sudo openssl req -new -key /etc/pki/tls/private/carmen.key -out /etc/pki/tls/carmen.csr

Actualizamos /etc/apache2/sites-available/miservidor.conf <Directory /var/www/miservidor>
Options Indexes FollowSymLinks
AllowOverride All
Require all granted
SSLOptions +StdEnvVars +ExportCertData

```
  </Directory>

  # Directorio Ventas - solo para OU=Contabilidad
  <Directory "/var/www/miservidor/ventas">
    Require expr %{SSL_CLIENT_S_DN_OU} == "Contabilidad"
  </Directory>
```

sudo apachectl configtest
sudo systemctl restart httpd

Y exportamos las claves a .p12
openssl pkcs12 -export -out carmen.p12 -inkey carmen.key -in carmen.crt -certfile /etc/ssl/certs/cacert.pem

En mi Máquina Rocky (Principal):

Creamos la clave privada de Carmen Cabrera y añadimos permisos:

sudo openssl genrsa -out /etc/pki/tls/private/carmen.key 2048
sudo chmod 600 /etc/pki/tls/private/carmen.key

Creamos un nuevo certificado:
sudo openssl req -new -key /etc/pki/tls/private/carmen.key -out /etc/pki/tls/carmen.csr

Valores:
If you enter '.', the field will be left blank.

Country Name (2 letter code) [ES]:
State or Province Name (full name) [Madrid]:
Locality Name (eg, city) [Alcorcon]:
Organization Name (eg, company) [CA de Pruebas]:
Organizational Unit Name (eg, section) []:Contabilidad
Common Name (eg, your name or your server's hostname) []:Carmen Cabrera
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:

Ver que en este caso no usamos Server_SAN, usamos usr_cert con los valores expuestos anteriormente:

```
sudo openssl ca -config /etc/pki/tls/openssl.cnf \
  -in /etc/pki/tls/carmen.csr \
  -out /etc/pki/tls/certs/carmen.crt \
  -days 825 \
  -extensions usr_cert
```

Comprobación:

```
sudo openssl x509 -in /etc/pki/tls/certs/carmen.crt -noout -text
```

Creamos el certificado personal para formato Windows

```
sudo openssl pkcs12 -export \
  -in /etc/pki/tls/certs/carmen.crt \
  -inkey /etc/pki/tls/private/carmen.key \
  -out /etc/pki/tls/certs/carmen.p12 \
  -name "Carmen Cabrera"
```

Actualizamos /etc/httpd/conf.d/miservidor.conf

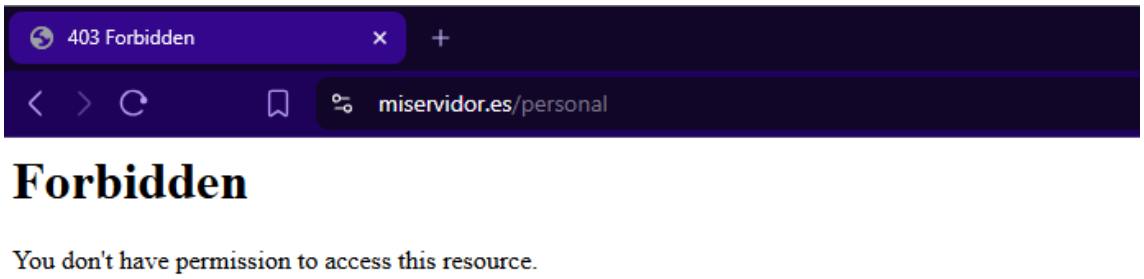
```
    <Directory /var/www/miservidor>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
        SSLOptions +StdEnvVars +ExportCertData
    </Directory>

    # Directorio Ventas - solo para OU=Contabilidad
    <Directory "/var/www/miservidor/ventas">
        Require expr %{SSL_CLIENT_S_DN_OU} == "Contabilidad"
    </Directory>
```

```
sudo apachectl configtest
sudo systemctl restart httpd
```

Ahora lo añadiremos a Windows de forma similar:

1) scp root@172.22.134:/etc/pki/tls/certs/carmen.p12 "C:\Users\Usuario\Downloads"
2) Volvemos a la herramienta de certificados de Windows
3) Esta vez lo importamos en la carpeta "Personal"

6.Crear un nuevo certificado a nombre de Mario Martinez (CN), del departamento de Marketing (OU), e instalarlo en el navegador. En el servidor web, crear dos directorios, Ventas y Personal, con una página de inicio en cada uno. Configurar el servidor Apache para que sólo permita el acceso a estos directorios a los miembros de los respectivos departamentos (campo OU), tras la autentificación mediante certificado al entrar en el sitio web.



Ambos https://www.miservidor.es
Carmen https://www.miservidor.es/ventas
Mario https://www.miservidor.es/personal

Una vez creados los certificados, las rutas y los índices procedemos con la configuración del Apache.

En Ubuntu:

Editamos el archivo de la pagina que ya teníamos habilitada previamente con el siguiente comando

vim /etc/apache2/sites-available/miservidor.conf

El documento quedará de la siguiente forma:

```
<VirtualHost *:80>
   ServerName www.miservidor.es
</VirtualHost>

<VirtualHost *:443>
   ServerName www.miservidor.com
   ServerAlias miservidor.com
   DocumentRoot /projects/miservidor
```

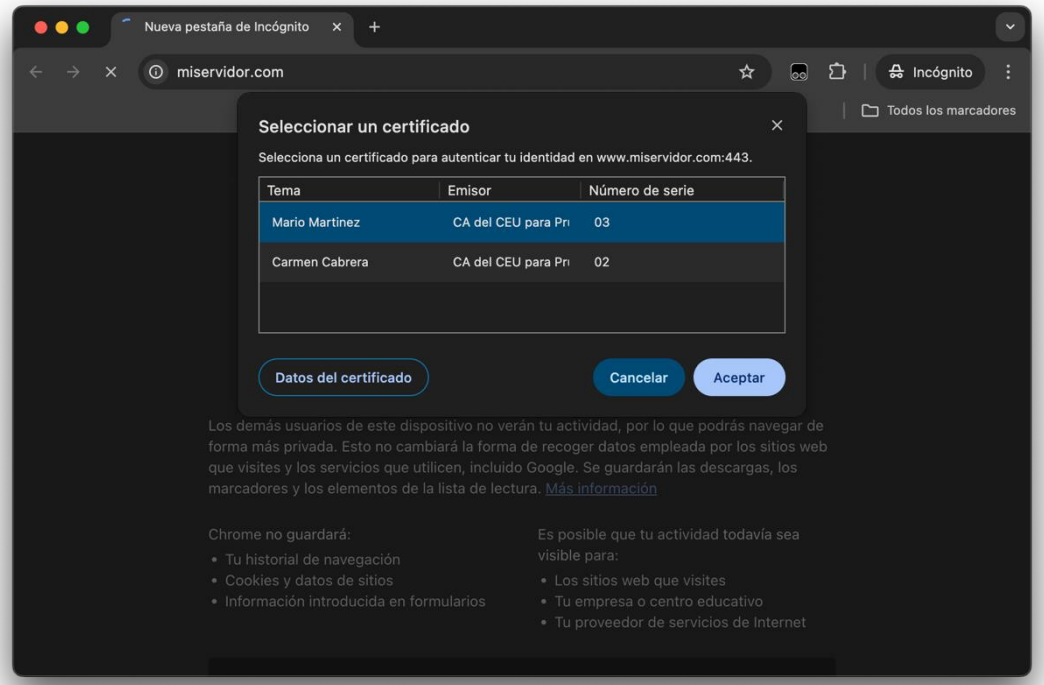
```
  SSLEngine on
  SSLCertificateFile /etc/pki/tls/server-crt.pem
  SSLCertificateKeyFile /etc/pki/tls/server-key.pem
  #SSLCertificateChainFile /etc/pki/tls/cacert.pem
  SSLCACertificateFile /etc/pki/tls/cacert.pem


  SSLVerifyClient require
  #<Location />
  #   SSLRequire ( %{SSL_CLIENT_S_DN_O} eq "CA de Pruebas" \
  #   && %{SSL_CLIENT_S_DN_CN} eq "Carmen Cabrera" \
  #   && %{SSL_CLIENT_S_DN_OU} eq "Departamento de contabilidad" )
  #</Location>

  <Directory /projects/miservidor>
    Options Indexes FollowSymLinks
    AllowOverride All
    Require all granted
  </Directory>
  <Directory /projects/miservidor/Ventas>
    Options Indexes FollowSymLinks
    AllowOverride All
    Require all granted
    SSLRequire %{SSL_CLIENT_VERIFY} eq "SUCCESS" \
    && %{SSL_CLIENT_S_DN_OU} eq "Departamento de contabilidad"
  </Directory>

  <Directory /projects/miservidor/Personal>
    Options Indexes FollowSymLinks
    AllowOverride All
    Require all granted
    SSLRequire %{SSL_CLIENT_VERIFY} eq "SUCCESS" \
    && %{SSL_CLIENT_S_DN_OU} eq "Departamento de Marketing"
  </Directory>
</VirtualHost>
```

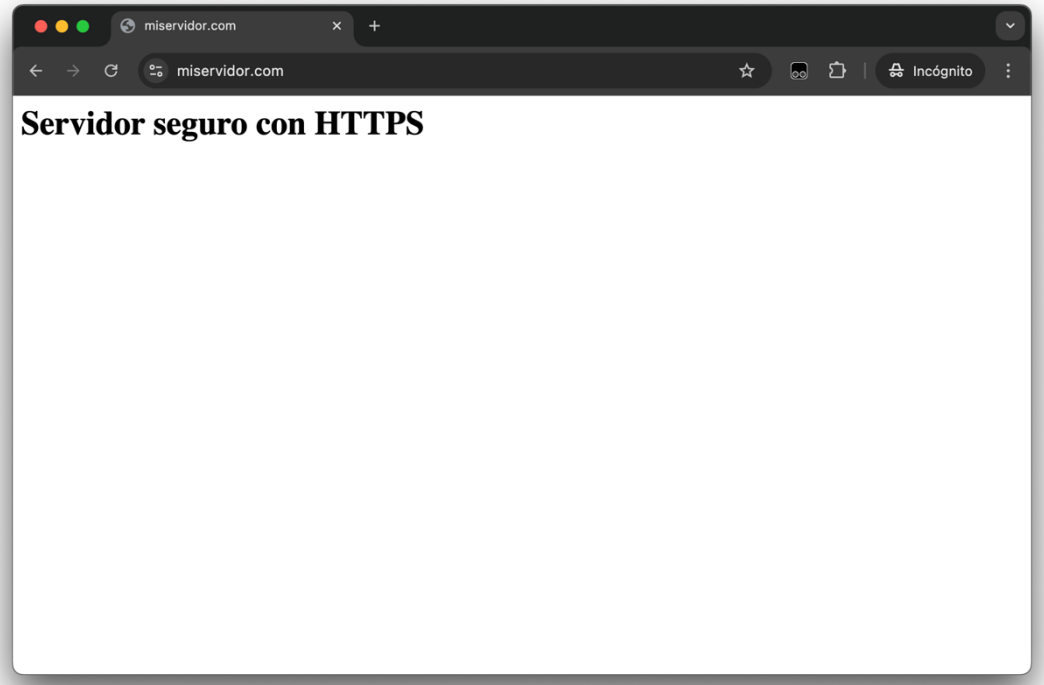
Y para recargar la página con los cambios se realiza con el siguiente comando:

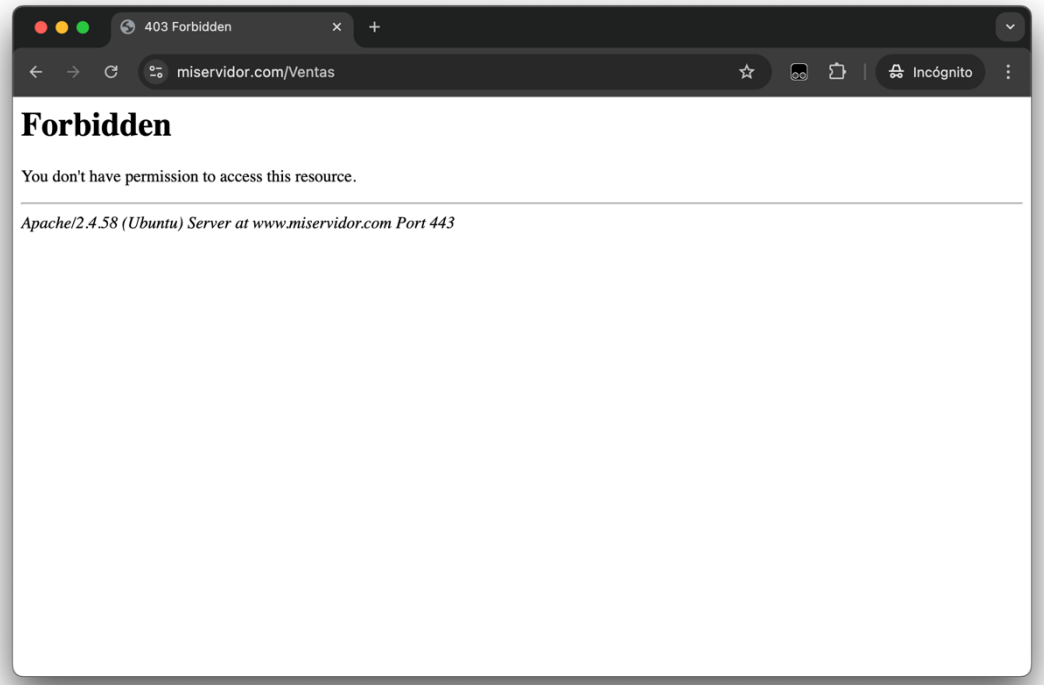
```
sudo systemctl reload apache2
```

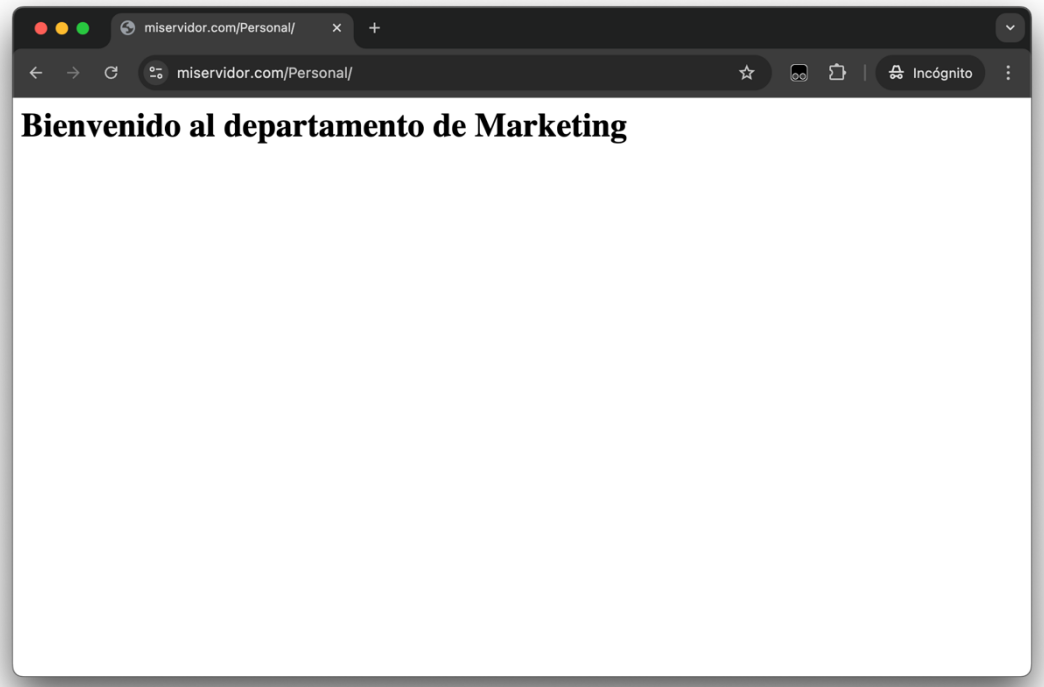
A continuación, se comprueba su funcionamiento a través del navegador

Al entrar con el certificado de Mario:

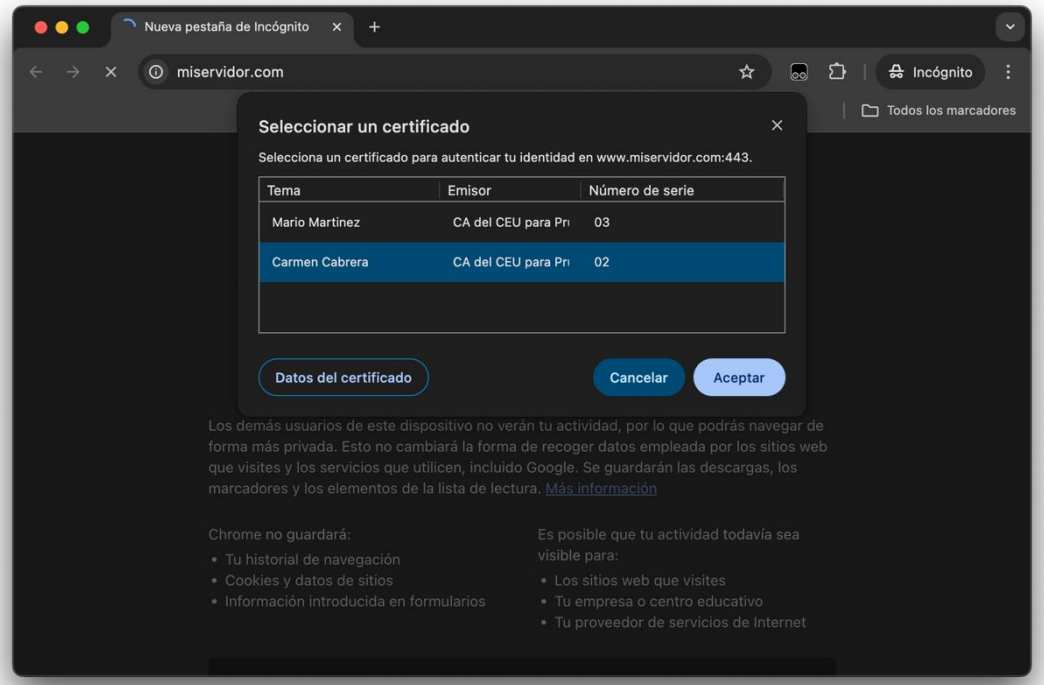

Nueva pestaña de Incógnito
miservidor.com
Incógnito
Todos los marcadores
Seleccionar un certificado
Selecciona un certificado para autenticar tu identidad en www.miservidor.com:443.
Tema
Emisor
Número de serie
Mario Martinez
CA del CEU para Pri
03
Carmen Cabrera
CA del CEU para Pri
02
Datos del certificado
Cancelar
Aceptar
Los demás usuarios de este dispositivo no verán tu actividad, por lo que podrás navegar de forma más privada. Esto no cambiará la forma de recoger datos empleada por los sitios web que visites y los servicios que utilicen, incluido Google. Se guardarán las descargas, los marcadores y los elementos de la lista de lectura. Más información
Chrome no guardará:
Tu historial de navegación
Cookies y datos de sitios
Información introducida en formularios
Es posible que tu actividad todavía sea visible para:
Los sitios web que visites
Tu empresa o centro educativo
Tu proveedor de servicios de Internet

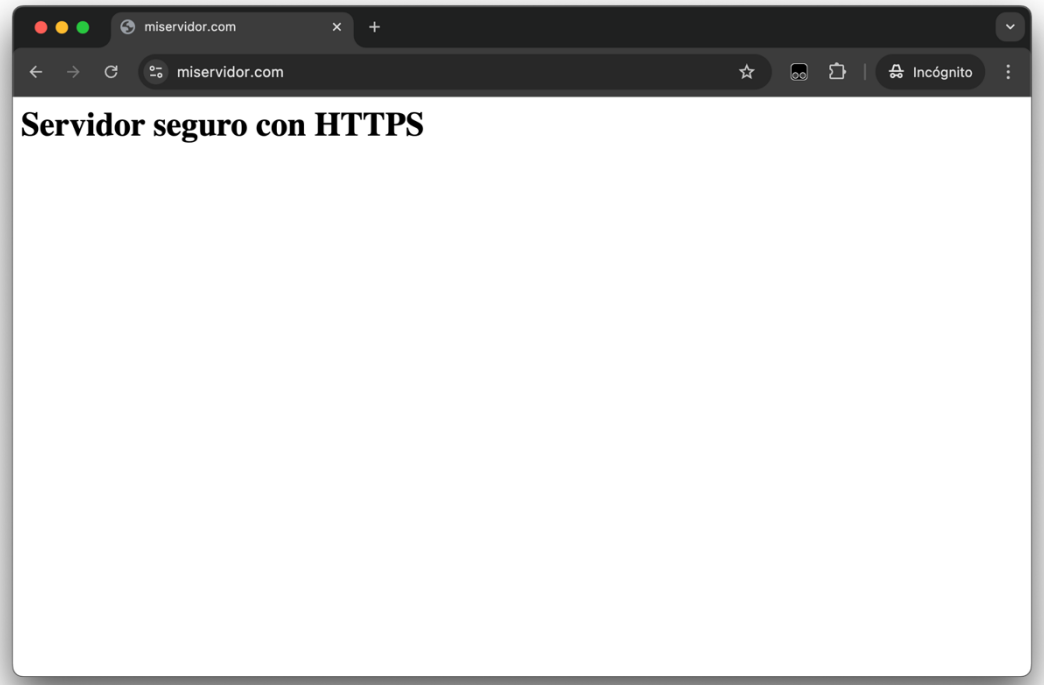

miservidor.com
miservidor.com
Incógnito
Servidor seguro con HTTPS



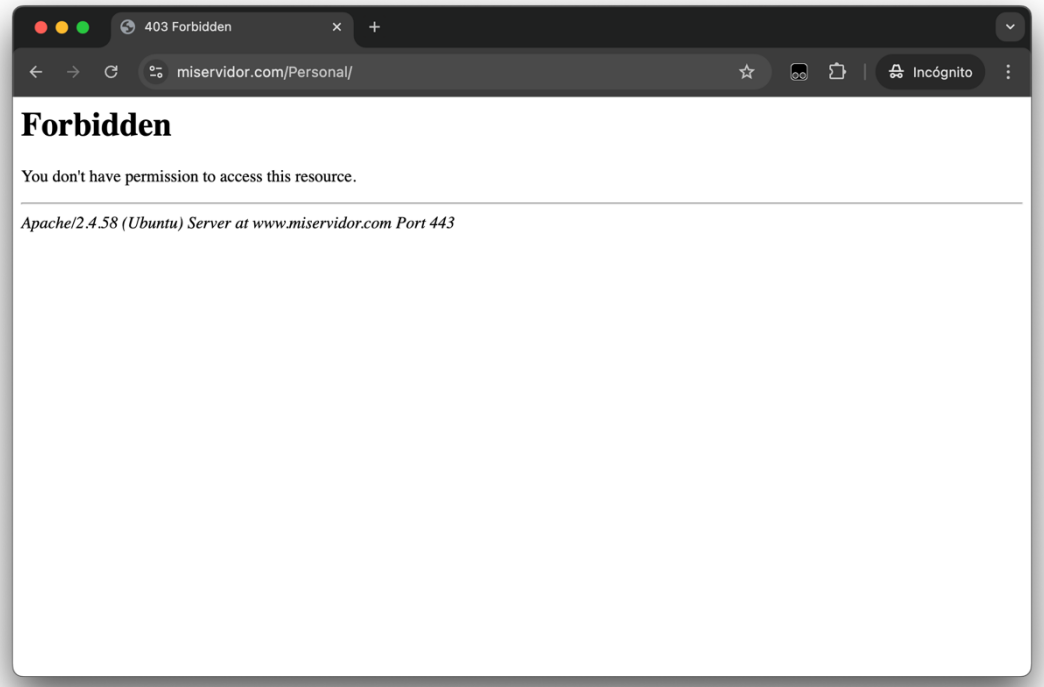


Si accedemos con el certificado de Carmen Cabrera:


Nueva pestaña de Incógnito
miservidor.com
Incógnito
Todos los marcadores
Seleccionar un certificado
Selecciona un certificado para autenticar tu identidad en www.miservidor.com:443.
Tema
Emisor
Número de serie
Mario Martinez
CA del CEU para Pri
03
Carmen Cabrera
CA del CEU para Pri
02
Datos del certificado
Cancelar
Aceptar
Los demás usuarios de este dispositivo no verán tu actividad, por lo que podrás navegar de forma más privada. Esto no cambiará la forma de recoger datos empleada por los sitios web que visites y los servicios que utilicen, incluido Google. Se guardarán las descargas, los marcadores y los elementos de la lista de lectura. Más información
Chrome no guardará:
Tu historial de navegación
Cookies y datos de sitios
Información introducida en formularios
Es posible que tu actividad todavía sea visible para:
Los sitios web que visites
Tu empresa o centro educativo
Tu proveedor de servicios de Internet


miservidor.com
miservidor.com
Incógnito
Servidor seguro con HTTPS


miservidor.com/Ventas/
miservidor.com/Ventas/
Incógnito
Bienvenido al departamento de Ventas



En la máquina Rocky (Principal):

Creamos la clave privada de Mario y proporcionamos permisos:
sudo openssl genrsa -out /etc/pki/tls/private/mario.key 2048
sudo chmod 600 /etc/pki/tls/private/mario.key

Creamos un nuevo certificado:
sudo openssl req -new -key /etc/pki/tls/private/mario.key -out /etc/pki/tls/mario.csr

Valores:
If you enter '.', the field will be left blank.

Country Name (2 letter code) [ES]:
State or Province Name (full name) [Madrid]:
Locality Name (eg, city) [Alcorcon]:
Organization Name (eg, company) [CA de Pruebas]:
Organizational Unit Name (eg, section) []:Marketing
Common Name (eg, your name or your server's hostname) []:Mario Martínez
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request

A challenge password []:
An optional company name []:

Ver que en este caso no usamos Server_SAN, usamos usr_cert con los valores expuestos anteriormente:

```
sudo openssl ca -config /etc/pki/tls/openssl.cnf \
 -in /etc/pki/tls/mario.csr \
 -out /etc/pki/tls/certs/mario.crt \
 -days 825 \
 -extensions usr_cert
```

Creamos el certificado personal para formato Windows

```
sudo openssl pkcs12 -export \
 -in /etc/pki/tls/certs/mario.crt \
 -inkey /etc/pki/tls/private/mario.key \
 -out /etc/pki/tls/certs/mario.p12 \
 -name "Mario Martínez"
```

Ahora lo añadiremos a Windows de forma similar:
1) scp root@172.22.134:/etc/pki/tls/certs/mario.p12 "C:\Users\Usuario\Downloads"
2) Volvemos a la herramienta de certificados de Windows
3) Esta vez lo importamos en la carpeta "Personal"

Creamos los departamentos de ventas y personal, y añadimos contenido:

```
sudo mkdir -p /var/www/miservidor/ventas
sudo mkdir -p /var/www/miservidor/personal
sudo chown -R apache:apache /var/www/miservidor

echo "<h1>Departamento de Ventas - Solo para Contabilidad</h1>" | sudo tee /var/www/miservidor/ventas/index.html
<h1>Departamento de Ventas - Solo para Contabilidad</h1>

echo "<h1>Departamento de Personal - Solo para Marketing</h1>" | sudo tee /var/www/miservidor/personal/index.html
<h1>Departamento de Personal - Solo para Marketing</h1>
```

Modificamos /etc/httpd/conf.d/miservidor.conf

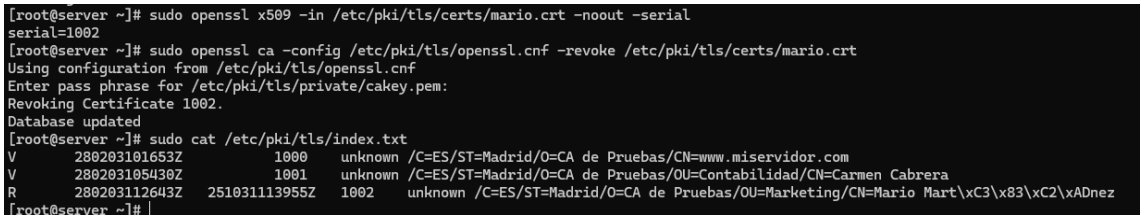
```
 <VirtualHost *:443>
    ServerName www.miservidor.es
    ServerAlias miservidor.es
```

```
    DocumentRoot /var/www/miservidor

    SSLEngine on
    SSLCertificateFile /etc/pki/tls/certs/miservidor.crt
    SSLCertificateKeyFile /etc/pki/tls/private/miservidor.key
    SSLCACertificateFile /etc/pki/tls/certs/cacert.pem

    # === NUEVAS DIRECTIVAS PARA AUTENTICACIÓN CLIENTE ===
    SSLVerifyClient require
    SSLVerifyDepth 2
    SSLOptions +StdEnvVars +ExportCertData
    # ================================================== #

    <Directory /var/www/miservidor>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
        SSLOptions +StdEnvVars +ExportCertData
    </Directory>

    # Directorio Ventas - solo para OU=Contabilidad
    <Directory "/var/www/miservidor/ventas">
        Require expr %{SSL_CLIENT_S_DN_OU} == "Contabilidad"
    </Directory>

    # Directorio Personal - solo para OU=Marketing
    <Directory "/var/www/miservidor/personal">
        Require expr %{SSL_CLIENT_S_DN_OU} == "Marketing"
    </Directory>
</VirtualHost>
```

Comprobamos y aplicamos:

```
sudo apachectl configtest
sudo systemctl restart httpd
```

7. Revocar el certificado del cliente Mario Martinez, y actualizar el archivo de certificados revocados de la CA. Configurar Apache para leer el archivo de revocaciones y comprobar a continuación que el usuario revocado ya no puede acceder vía HTTPS a los contenidos del servidor web.



En primer lugar procedemos a revocar el certificado de Mario Martinez y actualizar el listado de certificados revocados con los siguientes comandos:

openssl ca -revoke mario-crt.pem
openssl ca -gencrl -out crl.pem

Ubuntu:

Posteriormente editamos el archivo */etc/apache2/sites-available/miservidor.conf* para añadir que compruebe si el certificado esta revocado con el siguiente comando:

vim /etc/apache2/sites-available/miservidor.conf

Quedando el archivo de la siguiente forma:

```
  <VirtualHost *:80>
  ServerName www.miservidor.es
</VirtualHost>

<VirtualHost *:443>
  ServerName www.miservidor.com
  ServerAlias miservidor.com
  DocumentRoot /projects/miservidor

  SSLEngine on
  SSLCertificateFile /etc/pki/tls/server-crt.pem
  SSLCertificateKeyFile /etc/pki/tls/server-key.pem
  #SSLCertificateChainFile /etc/pki/tls/cacert.pem
  SSLCACertificateFile /etc/pki/tls/cacert.pem
  SSLCARevocationFile /etc/pki/tls/crl.pem
  SSLCARevocationCheck chain
  SSLProtocol all -SSLv3 -TLSv1.3


  SSLVerifyClient require
  #<Location />
  #   SSLRequire ( %{SSL_CLIENT_S_DN_O} eq "CA de Pruebas" \
  #   && %{SSL_CLIENT_S_DN_CN} eq "Carmen Cabrera" \
  #   && %{SSL_CLIENT_S_DN_OU} eq "Departamento de contabilidad" )
  #</Location>
```

```
  <Directory /projects/miservidor>
    Options Indexes FollowSymLinks
    AllowOverride All
    Require all granted
  </Directory>
  <Directory /projects/miservidor/Ventas>
    Options Indexes FollowSymLinks
    AllowOverride All
    Require all granted
    SSLRequire %{SSL_CLIENT_VERIFY} eq "SUCCESS" \
    && %{SSL_CLIENT_S_DN_OU} eq "Departamento de contabilidad"
  </Directory>

  <Directory /projects/miservidor/Personal>
    Options Indexes FollowSymLinks
    AllowOverride All
    Require all granted
    SSLRequire %{SSL_CLIENT_VERIFY} eq "SUCCESS" \
    && %{SSL_CLIENT_S_DN_OU} eq "Departamento de Marketing"
  </Directory>
</VirtualHost>
```

En la Máquina Rocky (Principal):

Revocar el certificado a Mario Martínez:

Comprobamos el número Serial de Mario:

```
sudo openssl x509 -in /etc/pki/tls/certs/mario.crt -noout -serial
```

Revocamos el certificado:

```
sudo openssl ca -config /etc/pki/tls/openssl.cnf -revoke
/etc/pki/tls/certs/mario.crt
```

Verificamos si fue revocado con éxito:

```
sudo cat /etc/pki/tls/index.txt
```

Creamos una lista donde se incluyan los revocados CRL:

```
sudo openssl ca -config /etc/pki/tls/openssl.cnf -gencrl -out
/etc/pki/tls/crl/crl.pem
```

Visualizar (opcional):

```
sudo openssl crl -in /etc/pki/tls/crl/crl.pem -noout -text
```

Modificamos /etc/httpd/conf.d/miservidor.conf de la siguiente forma:

```
<VirtualHost *:443>
   ServerName www.miservidor.es
   ServerAlias miservidor.es
   DocumentRoot /var/www/miservidor

   SSLEngine on
   SSLCertificateFile /etc/pki/tls/certs/miservidor.crt
   SSLCertificateKeyFile /etc/pki/tls/private/miservidor.key
   SSLCACertificateFile /etc/pki/tls/certs/cacert.pem

   # === NUEVAS DIRECTIVAS PARA AUTENTICACIÓN CLIENTE ===
   SSLVerifyClient require
   SSLVerifyDepth 2
   SSLOptions +StdEnvVars +ExportCertData
   # ================================================ #

   # === CONFIGURACIÓN CORREGIDA DEL CRL ===
   SSLCARevocationFile /etc/pki/tls/crl/crl.pem
   SSLCARevocationCheck chain
   SSLCipherSuite HIGH:!aNULL:!MD5
   SSLProtocol all -SSLv3 -TLSv1 -TLSv1.1
   # ====================================
   <Directory /var/www/miservidor>
      Options Indexes FollowSymLinks
      AllowOverride All
      Require all granted
      SSLOptions +StdEnvVars +ExportCertData
   </Directory>

   # Directorio Ventas - solo para OU=Contabilidad
   <Directory "/var/www/miservidor/ventas">
      Require expr %{SSL_CLIENT_S_DN_OU} == "Contabilidad"
   </Directory>

   # Directorio Personal - solo para OU=Marketing
   <Directory "/var/www/miservidor/personal">
      Require expr %{SSL_CLIENT_S_DN_OU} == "Marketing"
   </Directory>
</VirtualHost>
```

Comprobamos y aplicamos:

sudo apachectl configtest
sudo systemctl restart httpd

EXTRA, con este Script se debería de automatizar el file CRL
NO PROBADO:

sudo tee /usr/local/bin/update-crl.sh > /dev/null << 'EOF'
#!/bin/bash
/usr/bin/openssl ca -config /etc/pki/tls/openssl.cnf -gencrl -out /etc/pki/tls/crl/crl.pem
systemctl reload httpd
EOF

sudo chmod +x /usr/local/bin/update-crl.sh

8. Vamos a configurar un nuevo servidor virtual HTTPS (se recomienda usar otra distribución Linux distinta). Mantendrá el dominio www.pruebas.com (con su propio DocumentRoot, en /projects/pruebas). Las peticiones podrán hacerse sin incluir el prefijo www, y también tendremos en cuanta que algunos usuarios prefieren usar el nombre de dominio www.pruebas.net (nuevamente no será necesario incluir el prefijo www). Llevar a cabo la configuración del servidor, que contará con su página de inicio. Comprobar que accedemos al contenido del servidor sin errores ni alertas usando cualquiera de los nombres propuestos

En primer lugar, descargamos en la nueva máquina lo necesario para ejecutar los comandos y lo habilitamos.

sudo dnf install httpd mod_ssl -y
dnf install openssl
sudo systemctl enable --now httpd
sudo firewall-cmd --permanent --add-service=https
sudo firewall-cmd --reload

Después creamos el directorio que se nos exige en el cual se ubicará el index.html con un mensaje de bienvenida.

sudo mkdir -p /projects/pruebas
echo "<h1>Bienvenido a www.pruebas.com</h1>" | sudo tee /projects/pruebas/index.html

Editamos el archivo de tal manera que queda asi:

vim /etc/httpd/conf.d/pruebas.conf

```
<VirtualHost *:443>
   ServerName www.pruebas.com
   ServerAlias pruebas.com www.pruebas.net pruebas.net
   DocumentRoot /projects/pruebas

   SSLEngine on
   SSLCertificateFile /etc/pki/tls/server-crt.pem
   SSLCertificateKeyFile /etc/pki/tls/server-key.pem
   SSLCertificateChainFile /etc/pki/tls/cacert.pem

   <Directory /projects/pruebas>
      Require all granted
      AllowOverride All
   </Directory>
</VirtualHost>
```

Generamos los certificados tanto los de la CA raíz
openssl genrsa -aes256 -out private/cakey.pem 2048
openssl req -new -key private/cakey.pem -out ca-csr.pem
openssl req -x509 -extensions v3_ca -in ca-csr.pem -out cacert.pem -key private/cakey.pem -days 3652

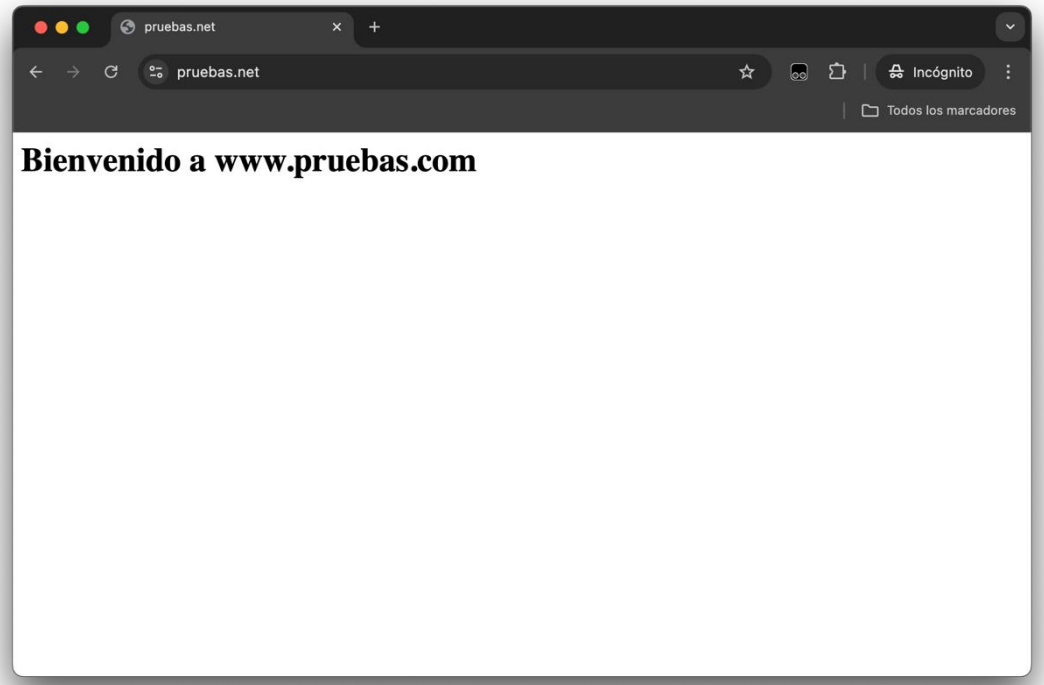
```
[root@localhost tls]# openssl req -new -key private/cakey.pem -out ca-csr.pem
Enter pass phrase for private/cakey.pem:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
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) [ES]:
State or Province Name (full name) [Madrid]:
Locality Name (eg, city) [Madrid]:
Organization Name (eg, company) [CA ap8]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

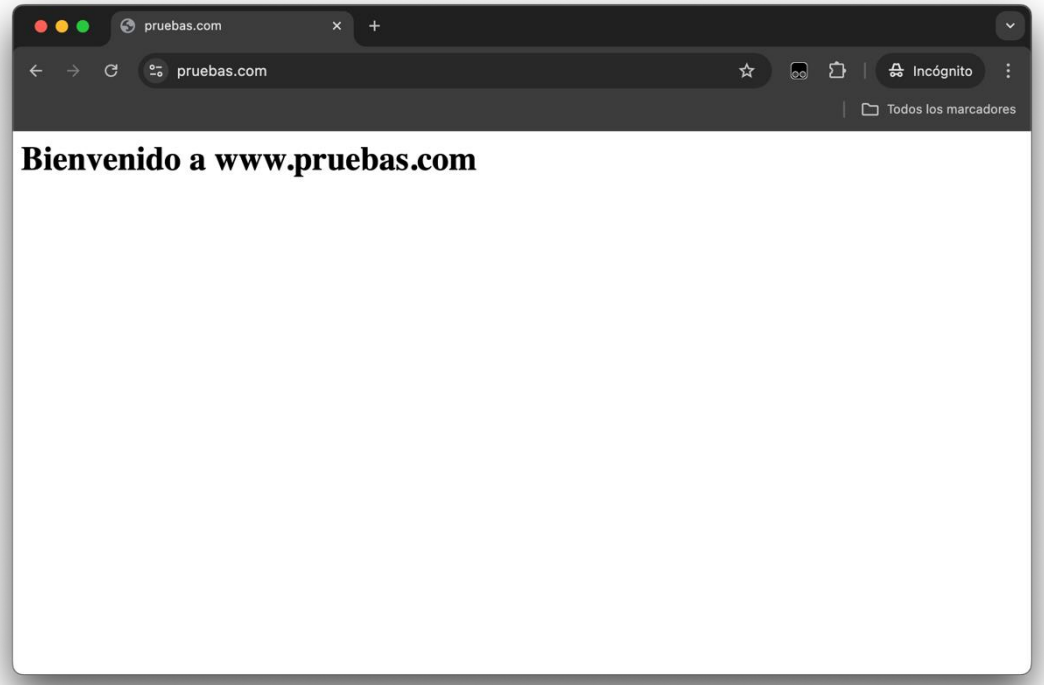
```
[root@localhost tls]# openssl req -x509 -extensions v3_ca -in ca-csr.pem -out cacert.pem -key private/cakey.pem -days 3652
Enter pass phrase for private/cakey.pem:
Warning: Not placing -key in cert or request since request is used
Warning: No -copy_extensions given; ignoring any extensions in the request
```

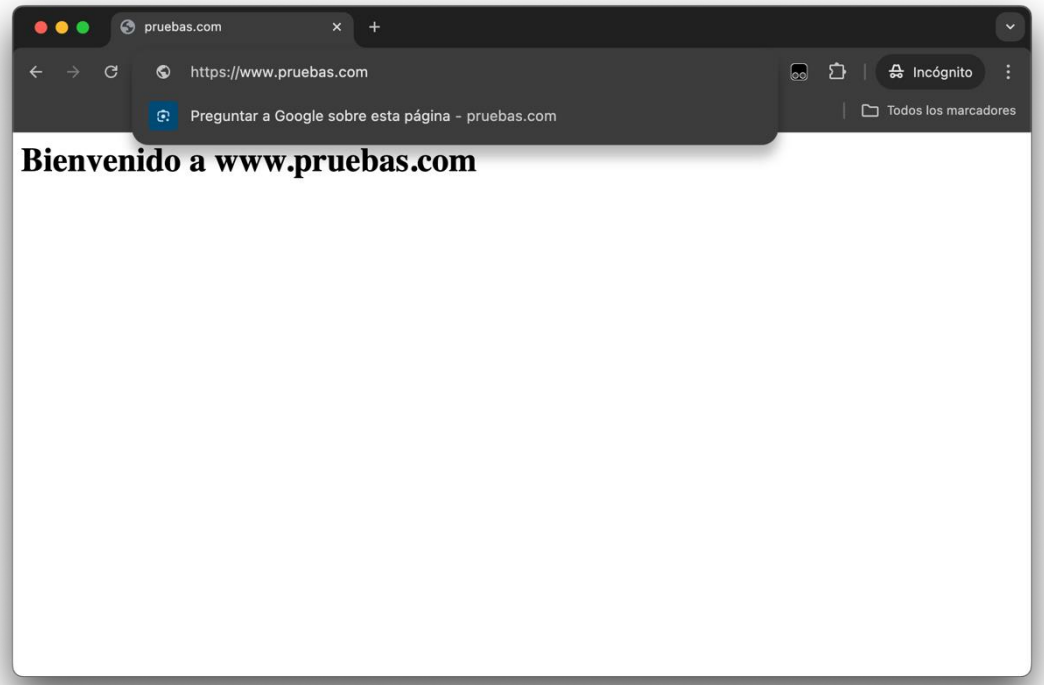
Como los del servidor:

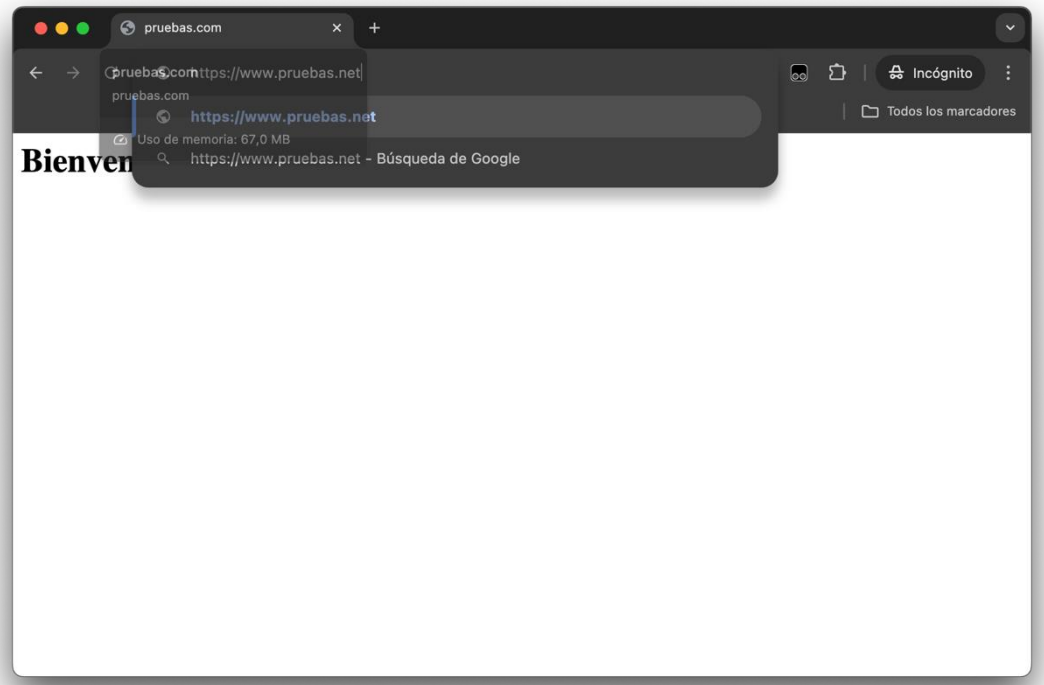
openssl genrsa -out server-key.pem 2048
openssl req -new -key server-key.pem -out server-csr.pem

```
-----
Country Name (2 letter code) [ES]:
State or Province Name (full name) [Madrid]:
Locality Name (eg, city) [Madrid]:
Organization Name (eg, company) [CA ap8]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) [CA ap8]:
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
[root@localhost tls]# openssl ca -extensions server_SAN -in server-csr.pem -out server-crt.pem -days 730
Using configuration from /etc/pki/tls/openssl.cnf
Enter pass phrase for /etc/pki/tls/private/cakey.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 1 (0x1)
        Validity
            Not Before: Nov  2 17:08:28 2025 GMT
            Not After : Nov  2 17:08:28 2027 GMT
        Subject:
            countryName               = ES
            stateOrProvinceName       = Madrid
            organizationName          = CA ap8
            commonName                = CA ap8
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Key Usage:
                Digital Signature, Non Repudiation, Key Encipherment
            X509v3 Extended Key Usage:
                TLS Web Server Authentication
            X509v3 Subject Alternative Name:
                DNS:www.pruebas.com, DNS:pruebas.com, DNS:www.pruebas.net, DNS:pruebas.net
```


pruebas.net
pruebas.net
Incógnito
Todos los marcadores
Bienvenido a www.pruebas.com


pruebas.com
pruebas.com
Incógnito
Todos los marcadores
Bienvenido a www.pruebas.com


pruebas.com
https://www.pruebas.com
Preguntar a Google sobre esta página - pruebas.com
Incógnito
Todos los marcadores
Bienvenido a www.pruebas.com


pruebas.com
pruebas.com
https://www.pruebas.net
pruebas.com
https://www.pruebas.net
Uso de memoria: 67,0 MB
https://www.pruebas.net - Búsqueda de Google
Incógnito
Todos los marcadores
Bienven

En la Máquina Rocky (Principal) y Ubuntu (Secundario)

En la máquina Rocky (La principal en este caso)

Vamos a crear los certificados desde la Máquina Rocky

Pruebas

```
sudo openssl genrsa -out /etc/pki/tls/private/pruebas.key 2048
sudo chmod 600 /etc/pki/tls/private/pruebas.key
```

```
sudo openssl req -new -key /etc/pki/tls/private/pruebas.key -out
/etc/pki/tls/pruebas.csr
```

```
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [ES]:
State or Province Name (full name) [Madrid]:
Locality Name (eg, city) [Alcorcon]:
Organization Name (eg, company) [CA de Pruebas]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname)
[]:www.pruebas.com
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

```
sudo openssl ca -config /etc/pki/tls/openssl.cnf \
  -in /etc/pki/tls/pruebas.csr \
  -out /etc/pki/tls/certs/pruebas.crt \
  -days 825 \
  -extensions server_SAN
```

Miempresa

```
sudo openssl genrsa -out /etc/pki/tls/private/wildcard_miempresa.key 2048
sudo chmod 600 /etc/pki/tls/private/wildcard_miempresa.key
```

```
sudo openssl req -new -key /etc/pki/tls/private/wildcard_miempresa.key -out
/etc/pki/tls/wildcard_miempresa.csr
```

```
If you enter '.', the field will be left blank.
-----
```

```
Country Name (2 letter code) [ES]:
State or Province Name (full name) [Madrid]:
Locality Name (eg, city) [Alcorcon]:
Organization Name (eg, company) [CA de Pruebas]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:*.miempresa.es
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

Vamos a crear este file temporal para cumplir con los requisitos que queremos que tenga el certificado:

```
sudo tee /tmp/wildcard.cnf > /dev/null << 'EOF'
[req]
distinguished_name = req_distinguished_name
req_extensions = v3_req

[req_distinguished_name]
countryName = ES
stateOrProvinceName = Madrid
localityName = Alcorcon
organizationName = CA de Pruebas
commonName = *.miempresa.es

[v3_req]
basicConstraints = CA:FALSE
keyUsage = digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth
subjectAltName = @alt_names

[alt_names]
DNS.1 = *.miempresa.es
DNS.2 = miempresa.es
EOF
```

```
sudo openssl ca -config /etc/pki/tls/openssl.cnf \
  -in /etc/pki/tls/wildcard_miempresa.csr \
  -out /etc/pki/tls/certs/wildcard_miempresa.crt \
  -days 825 \
  -extensions v3_req \
  -extfile /tmp/wildcard.cnf
```

Comprobar que tenemos los siguientes files creados:

```
sudo ls -la /etc/pki/tls/certs/pruebas.crt
sudo ls -la /etc/pki/tls/certs/wildcard_miempresa.crt
```

Preparamos una carpeta para luego copiar los certificados en la Ubuntu:

```
sudo mkdir /tmp/ubuntu-certs
sudo cp /etc/pki/tls/certs/cacert.pem /tmp/ubuntu-certs/
sudo cp /etc/pki/tls/certs/pruebas.crt /tmp/ubuntu-certs/
sudo cp /etc/pki/tls/private/pruebas.key /tmp/ubuntu-certs/
sudo cp /etc/pki/tls/certs/wildcard_miempresa.crt /tmp/ubuntu-certs/
sudo cp /etc/pki/tls/private/wildcard_miempresa.key /tmp/ubuntu-certs/
sudo chmod 644 /tmp/ubuntu-certs/*
```

Debemos tener todos estos archivos:

```
sudo ls -la /tmp/ubuntu-certs/
```

```
drwxr-xr-x  2 root root  122 nov  2 10:58 .
drwxrwxrwt. 17 root root 4096 nov  2 10:58 ..
-rw-r--r--  1 root root 2082 nov  2 10:58 cacert.pem
-rw-r--r--  1 root root 6390 nov  2 10:58 pruebas.crt
-rw-r--r--  1 root root 1704 nov  2 10:58 pruebas.key
-rw-r--r--  1 root root 5932 nov  2 10:58 wildcard_miempresa.crt
-rw-r--r--  1 root root 1704 nov  2 10:58 wildcard_miempresa.key
```

Enviamos los certificados a la máquina Ubuntu:

```
sudo scp -r /tmp/ubuntu-certs/* root@172.22.0.135:/tmp/
```

En la máquina Ubuntu:

Verificamos que tenemos los files:

```
ls /tmp/
```

```
cacert.pem
pruebas.crt
pruebas.key
wildcard_miempresa.crt
wildcard_miempresa.key
```

9. Suponga que ahora se decide usar ambos dominios (www.pruebas.com y www.pruebas.net) como dos proyectos independientes (con sus respectivos DocumentRoot y página de inicio). Configurar Apache para atender a ambos dominios. Nota: aunque es posible, en este apartado no usaremos nuevos certificados emitidos por nuestra CA; vamos a reutilizar el certificado creado para el apartado anterior.

En la Máquina Ubuntu (Secundaria o Auxiliar):

Creamos estos directorios y movemos/copiamos los files anteriores:

```
sudo mkdir -p /etc/ssl/certs
sudo mkdir -p /etc/ssl/private
sudo cp /tmp/pruebas.crt /etc/ssl/certs/
sudo cp /tmp/pruebas.key /etc/ssl/private/
sudo cp /tmp/wildcard_miempresa.crt /etc/ssl/certs/
sudo cp /tmp/wildcard_miempresa.key /etc/ssl/private/
sudo cp /tmp/cacert.pem /etc/ssl/certs/

sudo chmod 644 /etc/ssl/certs/*.crt /etc/ssl/certs/*.pem
sudo chmod 600 /etc/ssl/private/*.key
```

Instalamos los paquetes apache2 y openssl, luego los activamos

```
sudo apt update
sudo apt install apache2 openssl
sudo a2enmod ssl
sudo a2enmod rewrite
sudo systemctl start apache2
sudo systemctl enable apache2
sudo systemctl restart apache2
```

10. Crear un certificado wildcard para *.miempresa.es. Añadir un nuevo servidor virtual para proyecto1.miempresa.es, usando el nuevo certificado. Verificar que es posible acceder a los servicios de proyecto1.miempresa.es y www.miempresa.es usando el mismo certificado digital y sin errores.

Creamos un directorio para cada html:

```
sudo mkdir -p /var/www/pruebas
sudo mkdir -p /var/www/pruebas_com
sudo mkdir -p /var/www/pruebas_net
sudo mkdir -p /var/www/proyecto1
```

```
sudo mkdir -p /var/www/miempresa
```

Añadimos contenido rápidamente usando esto:

```
sudo echo "<h1>Bienvenido a Pruebas.com - Servido desde Ubuntu</h1>" |
sudo tee /var/www/pruebas/index.html
sudo echo "<h1>Sitio Oficial de Pruebas.COM</h1><p>Dominio .com para
proyectos comerciales</p>" | sudo tee /var/www/pruebas_com/index.html
sudo echo "<h1>Sitio Oficial de Pruebas.NET</h1><p>Dominio .net para
proyectos de red</p>" | sudo tee /var/www/pruebas_net/index.html
sudo echo "<h1>Proyecto 1 - miempresa.es</h1><p>Servido desde
Ubuntu</p>" | sudo tee /var/www/proyecto1/index.html
sudo echo "<h1>Portal Principal - miempresa.es</h1><p>Servido desde
Ubuntu</p>" | sudo tee /var/www/miempresa/index.html
```

Configuramos los VirtualHosts en Ubuntu:

/etc/apache2/sites-available/pruebas.conf

```
# Redirecciones HTTP a HTTPS para pruebas.com
<VirtualHost *:80>
   ServerName www.pruebas.com
   ServerAlias pruebas.com
   Redirect permanent / https://www.pruebas.com/
</VirtualHost>

<VirtualHost *:80>
   ServerName www.pruebas.net
   ServerAlias pruebas.net
   Redirect permanent / https://www.pruebas.net/
</VirtualHost>

# VirtualHost para pruebas.com
<VirtualHost *:443>
   ServerName www.pruebas.com
   ServerAlias pruebas.com
   DocumentRoot /var/www/pruebas_com

   SSLEngine on
   SSLCertificateFile /etc/ssl/certs/pruebas.crt
   SSLCertificateKeyFile /etc/ssl/private/pruebas.key
   SSLCACertificateFile /etc/ssl/certs/cacert.pem

   <Directory /var/www/pruebas_com>
      Options Indexes FollowSymLinks
      AllowOverride All
      Require all granted
```

```
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/pruebas_com_error.log
    CustomLog ${APACHE_LOG_DIR}/pruebas_com_access.log combined
</VirtualHost>

# VirtualHost para pruebas.net
<VirtualHost *:443>
    ServerName www.pruebas.net
    ServerAlias pruebas.net
    DocumentRoot /var/www/pruebas_net

    SSLEngine on
    SSLCertificateFile /etc/ssl/certs/pruebas.crt
    SSLCertificateKeyFile /etc/ssl/private/pruebas.key
    SSLCACertificateFile /etc/ssl/certs/cacert.pem

    <Directory /var/www/pruebas_net>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/pruebas_net_error.log
    CustomLog ${APACHE_LOG_DIR}/pruebas_net_access.log combined
</VirtualHost>
```

/etc/apache2/sites-available/miempresa.conf

```
# Redirecciones HTTP a HTTPS para miempresa.es
<VirtualHost *:80>
    ServerName proyecto1.miempresa.es
    Redirect permanent / https://proyecto1.miempresa.es/
</VirtualHost>

<VirtualHost *:80>
    ServerName www.miempresa.es
    ServerAlias miempresa.es
    Redirect permanent / https://www.miempresa.es/
</VirtualHost>

# VirtualHost para proyecto1.miempresa.es
<VirtualHost *:443>
    ServerName proyecto1.miempresa.es
    DocumentRoot /var/www/proyecto1
```

```
    SSLEngine on
    SSLCertificateFile /etc/ssl/certs/wildcard_miempresa.crt
    SSLCertificateKeyFile /etc/ssl/private/wildcard_miempresa.key
    SSLCACertificateFile /etc/ssl/certs/cacert.pem

    <Directory /var/www/proyecto1>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/proyecto1_error.log
    CustomLog ${APACHE_LOG_DIR}/proyecto1_access.log combined
</VirtualHost>

# VirtualHost para www.miempresa.es
<VirtualHost *:443>
    ServerName www.miempresa.es
    ServerAlias miempresa.es
    DocumentRoot /var/www/miempresa

    SSLEngine on
    SSLCertificateFile /etc/ssl/certs/wildcard_miempresa.crt
    SSLCertificateKeyFile /etc/ssl/private/wildcard_miempresa.key
    SSLCACertificateFile /etc/ssl/certs/cacert.pem

    <Directory /var/www/miempresa>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/miempresa_error.log
    CustomLog ${APACHE_LOG_DIR}/miempresa_access.log combined
</VirtualHost>
```

En /etc/apache2/apache2.conf añadimos "ServerName server.local" esto al final, sino ejecutar :

```
sudo tee /etc/apache2/conf-available/servername.conf > /dev/null <<EOF
ServerName server.local
EOF
```

Finalmente habilitamos pruebas y miempresa, validamos sintaxis y aplicamos.

```
sudo a2ensite pruebas.conf
sudo a2ensite miempresa.conf
sudo apache2ctl configtest
sudo systemctl restart apache2
```

En /etc/hosts añadimos los dominios:

```
127.0.0.1 www.pruebas.com pruebas.com
127.0.0.1 www.pruebas.net pruebas.net
127.0.0.1 proyecto1.miempresa.es
127.0.0.1 www.miempresa.es miempresa.es
```

Para más información (Opcional):

```
sudo openssl x509 -in /etc/ssl/certs/wildcard_miempresa.crt -noout -text |
grep -A5 "Subject Alternative Name"
sudo apache2ctl -S
sudo openssl x509 -in /etc/ssl/certs/wildcard_miempresa.crt -noout -text |
grep -A10 "X509v3"
```

Comenzamos a preparar el certificado para exportarlo:

```
sudo cp /etc/ssl/certs/cacert.pem /var/www/html/cacert2.pem
sudo chmod 644 /var/www/html/cacert2.pem

sudo cp /etc/ssl/certs/cacert.pem /root/cacert2.cer
sudo chmod 644 /root/cacert2.cer
```

Reiniciamos Apache para aplicar, y Habilitamos por si acaso

```
Sudo systemctl restart apache2
sudo a2ensite pruebas.conf // Por si acaso, no debería hacer falta volver a
ejecutarlo
sudo a2ensite miempresa.conf // Por si acaso, no debería hacer falta volver a
ejecutarlo
```

Ahora usando PowerShell volvemos a importar el certificado:

Ahora lo añadiremos a Windows de forma similar:
1) scp root@172.22.135:/root/cacert2.cer "C:\Users\Usuario\Downloads"
2) Volvemos a la herramienta de certificados de Windows
3) Esta vez lo importamos en la carpeta Entidades de certificación raíz de confianza

11. Firmar un documento PDF. Para realizar este apartado será necesario contar con una pareja de claves pública /privada y un certificado personal a nombre del alumno. Si es posible, se realizará con el DNI electrónico (se requiere un lector de SmartCard) o con un certificado personal emitido por

alguna CA reconocida (por ejemplo, la FNMT). Si no se cuenta con ninguno de estos certificados, también es posible usar un certificado a nuestro nombre emitido por nuestra propia CA (en este último caso, se deberá adjuntar en el envío el certificado raíz de esta CA). El documento PDF puede ser la propia memoria de la práctica, y tendrá la firma visible en la última página del documento.

Creamos nuestra CA

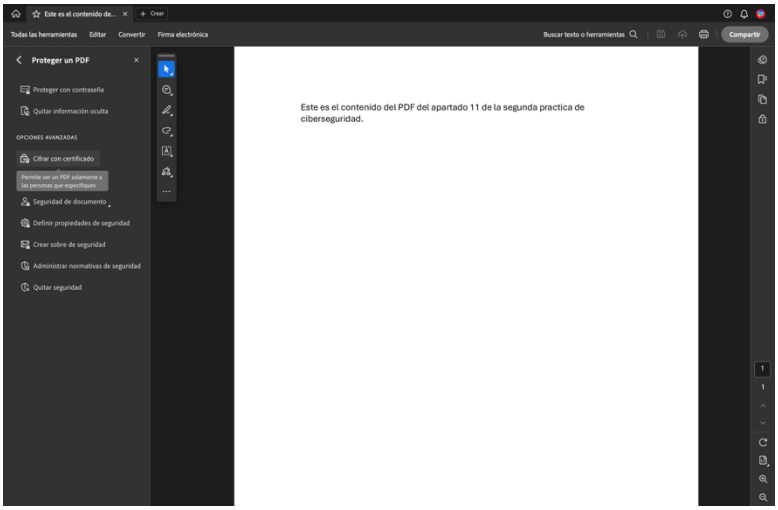
openssl genrsa -out ca.key 2048
openssl req -x509 -new -nodes -key ca.key -sha256 -days 1024 -out ca.pem

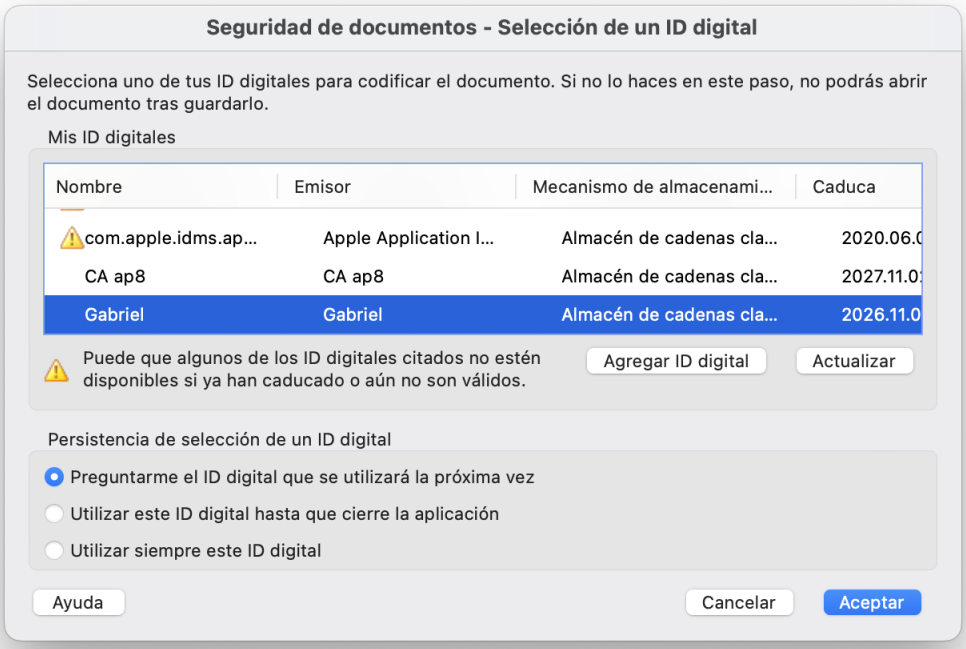
Despues el par de claves para el alumno y solicitar el certificado a la CA:

openssl genrsa -out alumno.key 2048
openssl req -new -key alumno.key -out alumno.csr
openssl x509 -req -in alumno.csr -CA ca.pem -CAkey ca.key -CAcreateserial -out alumno.crt -days 365 -sha256

Luego firmamos digitalmente con Adobe Acrobat con la clave generada que feu exportada a .p12 con el siguiente comando:

ap11 gabriellazovsky$ openssl pkcs12 -export -out alumnoc.p12 -inkey alumno.key -in alumno.crt -certfile ca.pem





Una vez firmado ya se puede utilizar únicamente con el archivo alumnoc.p12 y la contraseña “root”

Parte opcional:

12.

- Crear una nueva PKI que cuente con una autoridad raíz y una autoridad intermedia derivada de esta, que se encargue de emitir los certificados.

Emitir un certificado para un servidor web firmado por la autoridad intermedia. Por último, crear un archivo .pem que contenga la cadena de certificación completa (certificado del dominio, autoridad intermedia y autoridad raíz).

Comenzamos creando la estructura de directorios:

```
sudo mkdir -p
/etc/pki/ca_jerarquica/{root,intermediate}/{certs,crl,newcerts,private,csr}

sudo chmod 700 /etc/pki/ca_jerarquica/root/private
sudo chmod 700 /etc/pki/ca_jerarquica/intermediate/private
```

En el directorio /etc/pki/ca_jerarquica/root/openssl.root.cnf

[ca] default_ca = CA_root

[CA_root] dir = /etc/pki/ca_jerarquica/root certs = $dir/certs crl_dir = $dir/crl new_certs_dir = $dir/newcerts database = $dir/index.txt serial = $dir/serial RANDFILE = $dir/private/.rand

private_key = $dir/private/ca.root.key.pem certificate = $dir/certs/ca.root.crt.pem

crlnumber = $dir/crlnumber crl = $dir/crl/ca.root.crl.pem

default_days = 375 default_crl_days = 30 default_md = sha256 preserve = no

policy = policy_strict

[policy_strict] countryName = match stateOrProvinceName = match organizationName = optional # ← Ahora es opcional organizationalUnitName = optional commonName = supplied emailAddress = optional

[req] default_bits = 4096 default_md = sha256 default_keyfile = privkey.pem distinguished_name = req_distinguished_name x509_extensions = v3_ca string_mask = utf8only prompt = no

[req_distinguished_name] countryName = ES stateOrProvinceName = Madrid localityName = Alcorcon

[v3_ca] subjectKeyIdentifier = hash authorityKeyIdentifier = keyid:always,issuer basicConstraints = critical, CA:true keyUsage = critical, digitalSignature, cRLSign, keyCertSign

[v3_intermediate_ca]

subjectKeyIdentifier = hash authorityKeyIdentifier = keyid:always,issuer basicConstraints = critical, CA:true, pathlen:0 keyUsage = critical, digitalSignature, cRLSign, keyCertSign nsCertType = sslCA, emailCA

Inicializamos la BBDD:

```
sudo touch /etc/pki/ca_jerarquica/root/index.txt
echo 1000 | sudo tee /etc/pki/ca_jerarquica/root/serial
echo 1000 | sudo tee /etc/pki/ca_jerarquica/intermediate/serial
echo 1000 | sudo tee /etc/pki/ca_jerarquica/root/crlnumber
echo 1000 | sudo tee /etc/pki/ca_jerarquica/intermediate/crlnumber
```

En el directorio /etc/pki/ca_jerarquica/root/openssl.root.cnf

[ca] default_ca = CA_intermediate

[CA_intermediate] dir = /etc/pki/ca_jerarquica/intermediate certs = $dir/certs crl_dir = $dir/crl new_certs_dir = $dir/newcerts database = $dir/index.txt serial = $dir/serial RANDFILE = $dir/private/.rand

private_key = $dir/private/ca.intermediate.key.pem certificate = $dir/certs/ca.intermediate.crt.pem

crlnumber = $dir/crlnumber crl = $dir/crl/ca.intermediate.crl.pem

default_days = 375 default_crl_days = 30 default_md = sha256 preserve = no

policy = policy_loose

[policy_loose] countryName = optional stateOrProvinceName = optional localityName = optional organizationName = optional organizationalUnitName = optional commonName = supplied emailAddress = optional

[req] default_bits = 4096 default_md = sha256 default_keyfile = privkey.pem distinguished_name = req_distinguished_name x509_extensions = v3_intermediate_ca string_mask = utf8only prompt = no

[req_distinguished_name] countryName = ES stateOrProvinceName = Madrid localityName = Alcorcon 0.organizationName = CEU CA Intermedia commonName = CEU Intermediate CA

[v3_intermediate_ca] subjectKeyIdentifier = hash authorityKeyIdentifier = keyid:always,issuer basicConstraints = critical, CA:true, pathlen:0 keyUsage = critical, digitalSignature, cRLSign, keyCertSign

[server_cert] basicConstraints = CA:FALSE nsCertType = server nsComment = "OpenSSL Generated Server Certificate" subjectKeyIdentifier = hash authorityKeyIdentifier = keyid,issuer:always keyUsage = critical, digitalSignature, keyEncipherment extendedKeyUsage = serverAuth

Generar CA raiz:

```
sudo openssl genrsa -aes256 -out
/etc/pki/ca_jerarquica/root/private/ca.root.key.pem 4096
```

Crear certificado autofirmado CA Raíz:

```
sudo openssl req -config /etc/pki/ca_jerarquica/root/openssl.root.cnf \
  -key /etc/pki/ca_jerarquica/root/private/ca.root.key.pem \
  -new -x509 -days 7300 -sha256 -extensions v3_ca \
  -out /etc/pki/ca_jerarquica/root/certs/ca.root.crt.pem
```

Generar CA Intermedia:

```
sudo openssl genrsa -aes256 -out
/etc/pki/ca_jerarquica/intermediate/private/ca.intermediate.key.pem 4096
```

Crear CSR CA Intermedia:

```
sudo openssl req -config
/etc/pki/ca_jerarquica/intermediate/openssl.intermediate.cnf \
  -new -sha256 \
  -key /etc/pki/ca_jerarquica/intermediate/private/ca.intermediate.key.pem \
  -out /etc/pki/ca_jerarquica/intermediate/csr/ca.intermediate.csr.pem \
  -subj "/C=ES/ST=Madrid/L=Alcorcon/O=CA de Pruebas/CN=CEU Intermediate CA"
```

Firmar CA Intermedia con CA Raíz:

```
sudo openssl ca -config /etc/pki/ca_jerarquica/root/openssl.root.cnf \
  -extensions v3_intermediate_ca -days 3650 -notext -md sha256 \
  -in /etc/pki/ca_jerarquica/intermediate/csr/ca.intermediate.csr.pem \
  -out /etc/pki/ca_jerarquica/intermediate/certs/ca.intermediate.crt.pem
```

Generar clave privado servidor:

```
sudo openssl genrsa -out
/etc/pki/ca_jerarquica/intermediate/private/servidor.intermedio.key 2048
```

Crear CSR servidor:

```
sudo openssl req -new -sha256 \
  -key /etc/pki/ca_jerarquica/intermediate/private/servidor.intermedio.key \
  -out /etc/pki/ca_jerarquica/intermediate/csr/servidor.intermedio.csr \
  -subj "/C=ES/ST=Madrid/L=Alcorcon/O=CA de Pruebas/CN=servidor.ceu.es"
```

Firmar certificado servidor con CA Intermedia:

```
sudo openssl ca -config
/etc/pki/ca_jerarquica/intermediate/openssl.intermediate.cnf \
```

```
  -extensions server_cert -days 825 -notext -md sha256 \
  -in /etc/pki/ca_jerarquica/intermediate/csr/servidor.intermedio.csr \
  -out /etc/pki/ca_jerarquica/intermediate/certs/servidor.intermedio.crt
```

Crear Cadena de Certificación:

```
sudo cat /etc/pki/ca_jerarquica/intermediate/certs/servidor.intermedio.crt \
         /etc/pki/ca_jerarquica/intermediate/certs/ca.intermediate.crt.pem \
         /etc/pki/ca_jerarquica/root/certs/ca.root.crt.pem > \
         /etc/pki/ca_jerarquica/cadena-completa.pem
```

Verificaciones:

```
sudo openssl verify -CAfile /etc/pki/ca_jerarquica/root/certs/ca.root.crt.pem \
  /etc/pki/ca_jerarquica/intermediate/certs/ca.intermediate.crt.pem

sudo openssl verify -CAfile /etc/pki/ca_jerarquica/cadena-completa.pem \
  /etc/pki/ca_jerarquica/intermediate/certs/servidor.intermedio.crt



sudo openssl x509 -in
/etc/pki/ca_jerarquica/intermediate/certs/servidor.intermedio.crt -noout -issuer
-subject -dates
sudo openssl x509 -in
/etc/pki/ca_jerarquica/intermediate/certs/ca.intermediate.crt.pem -noout -issuer
-subject -dates
sudo openssl x509 -in /etc/pki/ca_jerarquica/root/certs/ca.root.crt.pem -noout -
issuer -subject -dates
```

```
[root@server ~]# sudo tree /etc/pki/ca_jerarquica/
/etc/pki/ca_jerarquica/
├── cadena-completa.pem
├── intermediate
│   ├── certs
│   │   ├── cadena-completa.pem
│   │   ├── ca.intermediate.crt.pem
│   │   └── servidor.intermedio.crt
│   ├── crl
│   ├── crlnumber
│   ├── csr
│   │   ├── ca.intermediate.csr.pem
│   │   └── servidor.intermedio.csr
│   ├── index.txt
│   ├── index.txt.attr
│   ├── index.txt.old
│   ├── newcerts
│   │   └── 1000.pem
│   ├── openssl.intermediate.cnf
│   ├── private
│   │   ├── ca.intermediate.key.pem
│   │   └── servidor.intermedio.key
│   ├── serial
│   └── serial.old
└── root
    ├── certs
    │   └── ca.root.crt.pem
    ├── crl
    ├── crlnumber
    ├── csr
    ├── index.txt
    ├── index.txt.attr
    ├── index.txt.old
    ├── newcerts
    │   └── 1000.pem
    ├── openssl.root.cnf
    ├── private
    │   └── ca.root.key.pem
    ├── serial
    └── serial.old
```

- Con el fin de promover el uso de conexiones HTTPS de forma generalizada, la entidad certificadora Let's Encrypt (https://letsencrypt.org/) ofrece desde 2016 certificados gratuitos reconocidos por la mayor parte de los navegadores. Estudiar y documentar el proceso que debe seguirse para la obtención de estos certificados y su renovación periódica (es necesario tener registrado el dominio).

Como no poseemos un dominio (registrado y accesible por Internet), usaremos uno ficticio pero enseñaremos la ejecución de comandos.

```
# Instalación de Certbot
sudo apt update
sudo apt install certbot python3-certbot-apache

# Obtención del certificado (requiere dominio real)
sudo certbot --apache -d midominio-real.com -d www.midominio-real.com

# Verificación
sudo certbot certificates
```

```
# Renovación manual
sudo certbot renew

# Renovación automática vía cron
sudo crontab -e
# Añadir: 0 12 * * * /usr/bin/certbot renew --quiet
```

Certbot es capaz de modificar el VirtualHost de forma automática y los certificados son válidos por 90 días.

- Seleccionar alguna herramienta opensource para desplegar y mantener una PKI (OpenXPKI, Dogtag, etc.). Instalar y probar la herramienta elegida.

Herramienta open source seleccionada: EJBCA + wildfly

Instalamos los siguientes paquetes:

```
sudo dnf install -y java-11-openjdk java-11-openjdk-devel mariadb-server
mariadb ant wget unzip git


# Conf Maria DB
sudo systemctl start mariadb
sudo systemctl enable mariadb
sudo mysql_secure_installation
```

Comenzamos a instalar EJBCA:

```
sudo mkdir -p /opt/ejbca
cd /opt/ejbca
git clone https://github.com/Keyfactor/ejbca-ce.git
sudo chown -R $(whoami):$(whoami) /opt/ejbca
cat > conf/database.properties << 'EOF'
database.name=mysql
database.url=jdbc:mysql://localhost:3306/ejbca
database.driver=com.mysql.cj.jdbc.Driver
database.username=ejbca
database.password=ejbca_password
EOF

cat > conf/ant.properties << 'EOF'
appserver.home=/opt/ejbca
appserver.type=standalone
install.dir=/opt/ejbca
appserver.jboss.home=/opt/ejbca
appserver.jboss.config=standalone
EOF

cat > conf/ejbca.properties << 'EOF'
ejbca.datasource.jndi-name=java:/EjbcaDS
```

```
ejbca.httpsserver.hostname=localhost
ejbca.httpsserver.port=8443
ejbca.httpsserver.https=true
EOF
```

Instalamos wildfly para poder build EJBCA:

```
Cd /opt
sudo wget
https://github.com/wildfly/wildfly/releases/download/26.1.3.Final/wildfly-26.1.3.Final.tar.gz
sudo tar -xzf wildfly-26.1.3.Final.tar.gz
sudo ln -s wildfly-26.1.3.Final wildfly
sudo chown -R $(whoami):$(whoami) /opt/wildfly-26.1.3.Final
sudo chown -R $(whoami):$(whoami) /opt/wildfly
export APPSRV_HOME=/opt/wildfly
```

```
cd /opt/ejbca/ejbca-ce

cat > conf/ant.properties << 'EOF'
appserver.home=/opt/wildfly
appserver.type=jboss
appserver.subtype=jboss7
install.dir=/opt/ejbca/ejbca-ce
appserver.jboss.home=/opt/wildfly
appserver.jboss.config=standalone
EOF

ant clean
ant build
# Ponemos todo como default
ls -la dist/ejbca.ear
```

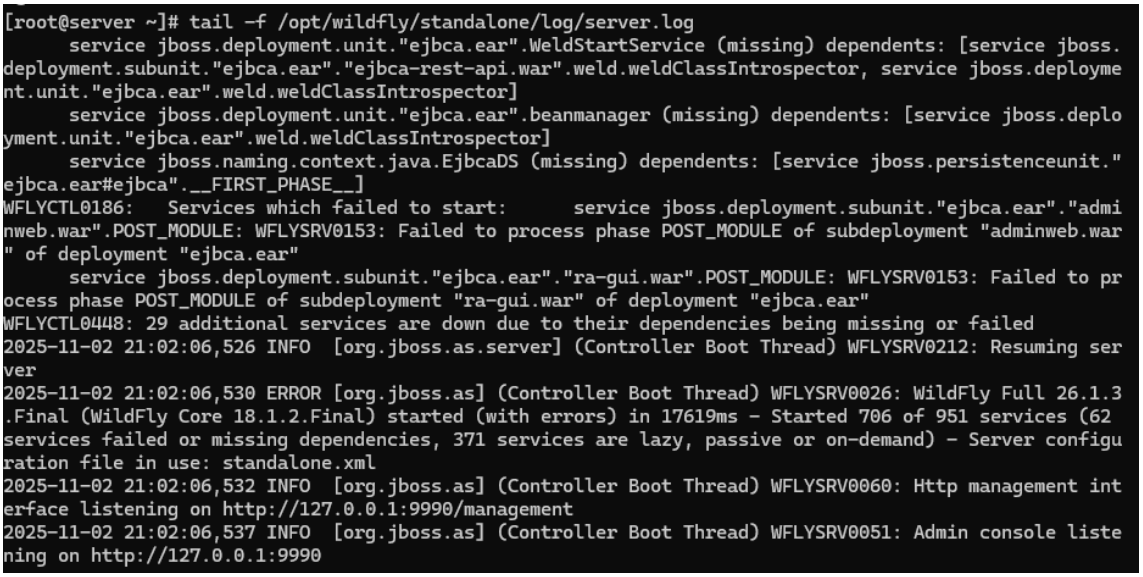
Iniciar EJBCA y wildfly:

```
cd /opt/ejbca/ejbca-ce
ant deployear

# En otra terminal

cd /opt/wildfly
./bin/standalone.sh -b 0.0.0.0
```

```
[root@server ~]# netstat -tlnp | grep -E '(8443|8442|8080|9990)'
tcp        0      0 0.0.0.0:8080            0.0.0.0:*               LISTEN      6817/java
tcp        0      0 127.0.0.1:9990          0.0.0.0:*               LISTEN      6817/java
tcp        0      0 0.0.0.0:8443            0.0.0.0:*               LISTEN      6817/java
```

Para ver logs:

```
tail -f /opt/wildfly/standalone/log/server.log | grep -i ejbca
tail -f /opt/wildfly/standalone/log/server.log
```



Para probar https://localhost:8443/ejbca/adminweb/