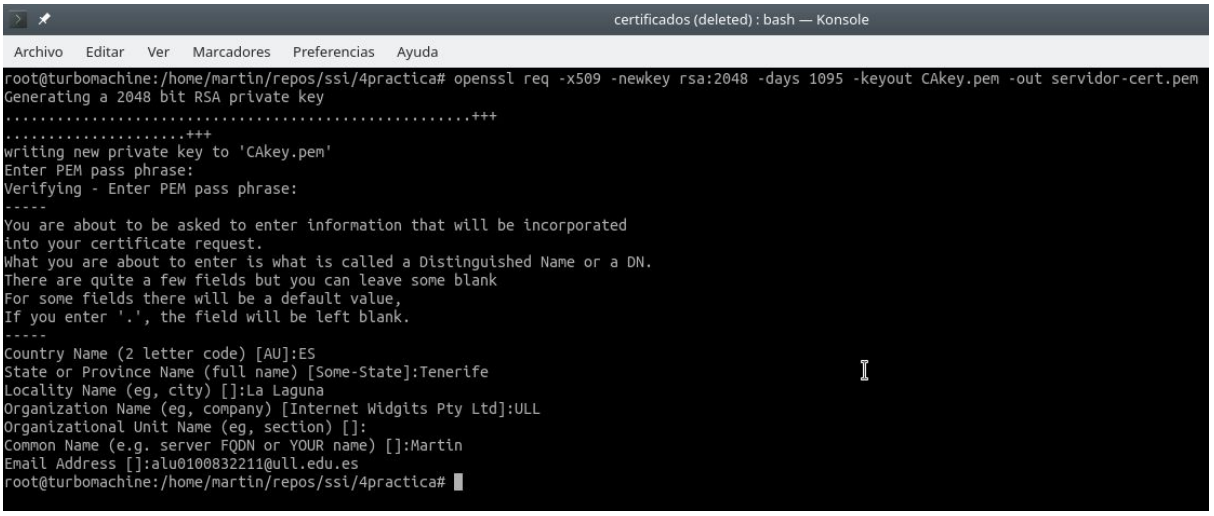
1.1 Generación de una autoridad certificadora

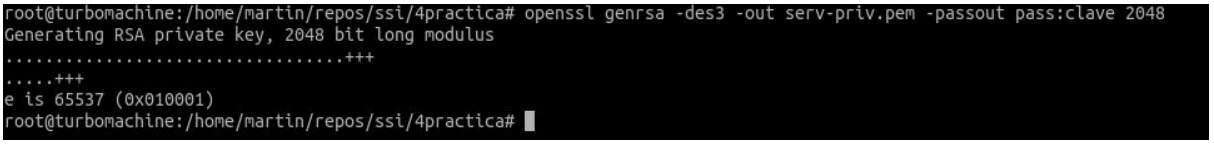
Se crea el certificado y la clave privada de la CA con el siguiente comando:



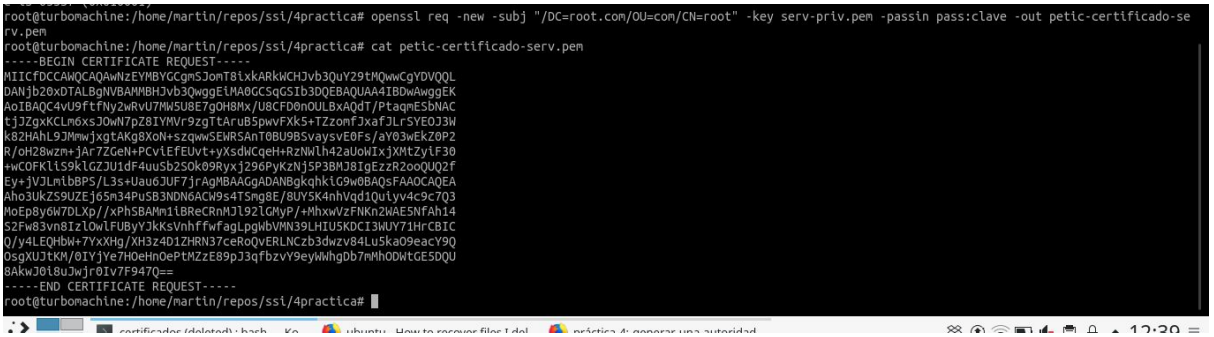
Se le está indicando que use una clave privada con RSA de 2048 bits para 1095 días.

1.2 Generación del certificado del servidor

En este paso se simula cómo un servidor pediría el certificado a una CA.
Se genera la clave privada del servidor. (Desde el servidor)



Con esa clave privada, se genera una petición de certificado que define al servidor como propietario. (Desde el servidor)



Con esa petición, la CA emite el certificado del servidor usando el certificado de la CA y la clave privada de la CA.

```
certificados git:(master) ✗ sudo openssl x509 -CA CAcert.pem -CAkey CAkey.pem -
req -in petic-certificado-serv.pem -days 15 -extfile config1.txt -sha1 -CAcreatese
rial -out servidor-cert.pem
[sudo] password for martin:
Signature ok
subject=DC = root.com, OU = com, CN = root
Getting CA Private Key
Enter pass phrase for CAkey.pem:
certificados git:(master) ✗ cat servidor-cert.pem
-----BEGIN CERTIFICATE-----
MIIDazCCAlOgAwIBAgIJAPZ4qCvLXrOrMA0GCSqGSIb3DQEBBQUAMIGLMQswCQYD
VQQGEwJFUzERMA8GA1UECAwIVGVuZXJpZmUxEjAQBgNVBAcMCUxhIExhZ3VuYTEM
MAoGA1UECgwDVUxMMQ0wCwYDVQQLDAREcHRvMQ8wDQYDVQQDDAZNYXJ0aW4xJzAl
BgkqhkiG9w0BCQEWGGFsdTAxMDA4MzIyMTFAdWxsLmVkdS5lczAeFw0xODExMDkx
MzUxNTRaFw0xODExMjQxMzUxNTRaMDcxGDAWBgoJkiaJk/IsZAEZFghyb290LmNv
bTEMMAoGA1UECwwDY29tMQ0wCwYDVQQDDARyb290MIIBIjANBgkqhkiG9w0BAQEF
AAOCAQ8AMIIBCgKCAQEAzrGrJPFcNaRhJMuvzEirtRQQPSQfY6womlB02EyQGmx2
3PtrxjZUWDmtAva7iAXG2l0Vh5GDGBF1VQnE4tRFHfU/NQu6kPZbkfol/J56pVLO
TcdZ4QO2ZZDQlGpiUJwDFbjXP5xFsgRgB+dKPyYmigAzamycUgOsaKiIDxQlV5+P
eXpO6UVU4OSuAqO+FTEM9lyeD2hPzgdwv/sAOBtUnzsxONWbKdAz5W9IhDDGALmn
ABFZBHo7MPIdC95Ac5wHFlLEXyvSiP0BLPBd1isbGT24kLUMbbwBNf5drw22qXFm
0mhV0J7YUTeIIa/I/iZJD5cii0bjaB9Og4jt1odnMQIDAQABoyUwIzAMBgNVHRMB
Af8EAjAAMBMGA1UdJQQMMAoGCCsGAQUFBwMBMA0GCSqGSIb3DQEBBQUAA4IBAQAL
xBBfMr/KvXJ3Hzoss9d8rQvcefCglPVqSOtJrAO9xFao3Wu43YNLSdZzPwZj7sg
5fKHUqZR8oIt1s1XE413YF5x2egzj2REoTCD/0LzkpSDj/PJzUsjIkfJSNRhaJuO
4kLOxf6VjfMi8YGxrwyOC/FidQR9GLtQtS4qNFgGtz7mkZUJ/D6Ijpze++avyfil
CfqomhikvgDXpLL590EJNimdjtdFybJp3on9v5PwuiEjcwAHn+dBa813qzcAYEMw
3JI87k2dbDFYnDuHgV9Qs9tN/kkwmU1icZf58jS8lMWxiZ4s6GwCCqnhSx/wbUHV
otbRP0n10/Aq0Jg+txw4
-----END CERTIFICATE-----
certificados git:(master) ✗
```

1.3 Generación de los certificados de los clientes

En este paso se simula la transacción entre un cliente y la CA para que el cliente obtenga un certificado que pueda usar en internet.
Se crea la clave privada del cliente (en el cliente).

```
root@turbomachine:/home/martin/repos/ssi/4practica/certificados# openssl genrsa -d
es3 -passout pass:clave -out client-priv.pem 2048
Generating RSA private key, 2048 bit long modulus
..............................+++
.....................................................................+++
e is 65537 (0x010001)
root@turbomachine:/home/martin/repos/ssi/4practica/certificados#
```

Con esa clave privada se crea una petición para obtener un certificado de la CA

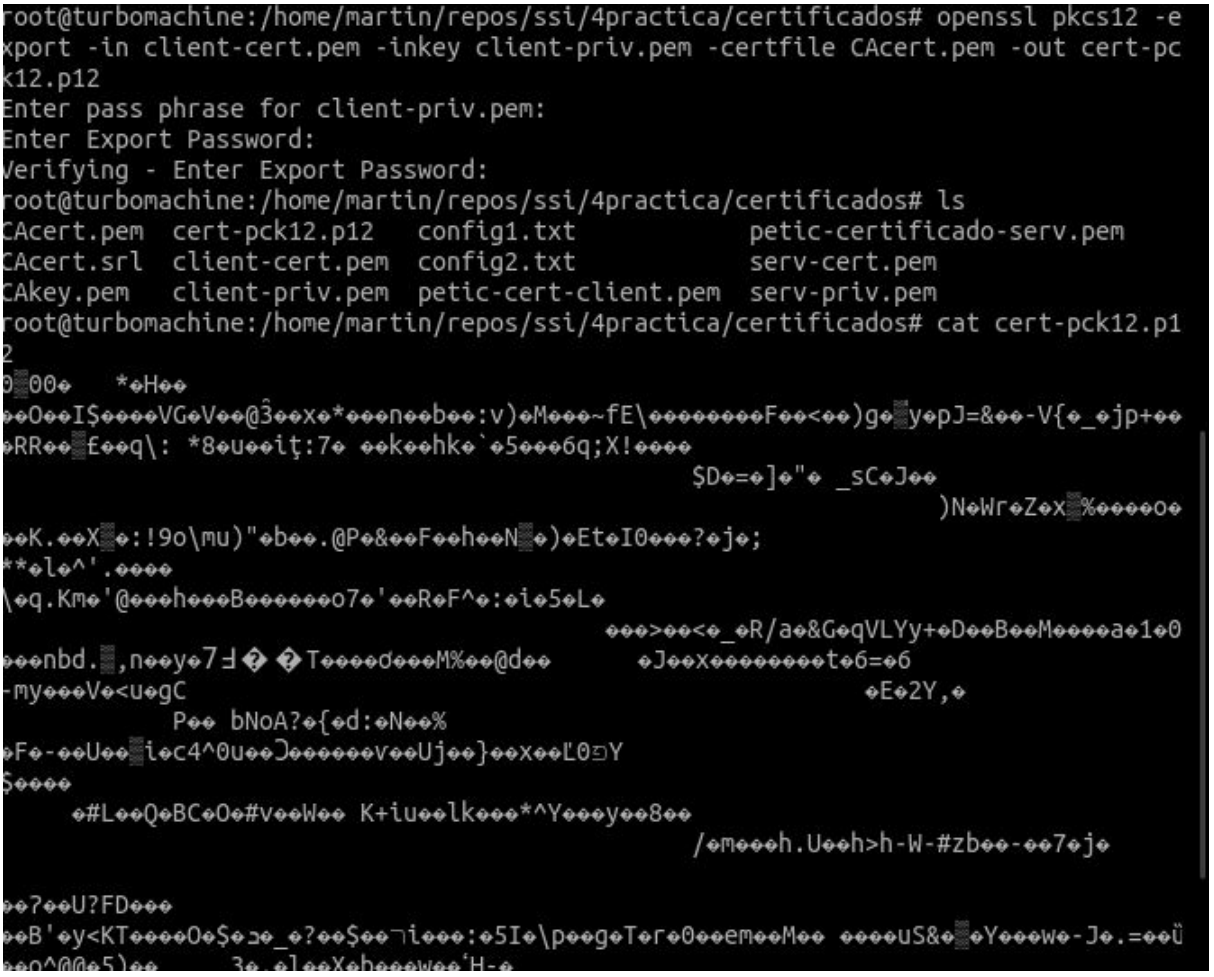
```
root@turbomachine:/home/martin/repos/ssi/4practica/certificados# openssl req -new
-key client-priv.pem -passin pass:clave -subj "/DC=localhost/OU=com/CN=Fsv" -out p
etic-cert-client.pem
root@turbomachine:/home/martin/repos/ssi/4practica/certificados# ls
CAcert.pem  client-priv.pem        petic-certificado-serv.pem
CAcert.srl  config1.txt            serv-cert.pem
CAkey.pem   petic-cert-client.pem  serv-priv.pem
root@turbomachine:/home/martin/repos/ssi/4practica/certificados# cat petic-cert
cat: petic-cert: No such file or directory
root@turbomachine:/home/martin/repos/ssi/4practica/certificados# cat petic-cert-cl
ient.pem
-----BEGIN CERTIFICATE REQUEST-----
MIICfDCCAWQCAQAwNzEZMBcGCgmSJomT8ixkARkWCWxvY2FsaG9zdDEMMAoGA1UE
CwwDY29tMQwwCgYDVQQDDANGc3YwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEK
AoIBAQDG6n32Jgi1wxvZVcdht+DHiLYW7oV9bfX8iI++BTs7mDlAE3l0ZErO1CcZ
xWttMTY1oYcb5XxJ7a7ncHkE8xXBD+QTL609mil5OMJPnH+lM0M1qX5QUE2qmkyu
2VCM6Oug6ogXd961jW1HLBUfxkpG5sWY2kqyl/cKvIsv8SVhhEM0b10owD4mXqz9
vmMEY1/7rglkNmbJUS4wBUZu2k/aR+vu4iZlLh+1uBxku6GX6HpAp2mXSfwAXVO5
mrD+6V//ukeRmNyDjtIfU8g0O5E0M0UdqWsRQDBKjccwWVaFwWrr6Tm6sgrsubOs
yJvbL+0H4lRJ4tXCWSg1YstAhbNTAgMBAAGgADANBgkqhkiG9w0BAQsFAAOCAQEA
fjbL/et2MAx8JJISPje16KgV8PJ0eG6R/6C3rLimmw1fleXZfyeONnH5O9XnLv7l
sGHlhDrTDt+SToc+/kT4ntRttbAW1UTalODu887inqp64RitskCEWIMY9gTbU4PD
Zom7IkhFrl+wfPgt9ViPifnVeNli796hGEjpfKhV49VdWl01ADp6iiZJo/Yw0y+t
3Lb+sLNZ8G+O4WGvcDAgYQUjDlyJ7Jg8LQhbNCL+J0nWdZOAD35EVxPnTVV1lr8+
1BsnT6/s50NEttUhFf99YYbmgBH/kPDjiD/oEmwYco7Xq1nYxjdAPd0NhOrjTP8
QDC6cINxuVtodIvMQ8Nh/Q==
-----END CERTIFICATE REQUEST-----
root@turbomachine:/home/martin/repos/ssi/4practica/certificados#
```

Con la petición, la CA usa su certificado y su clave privada para crear el certificado del cliente mediante el comando

```
openssl x509 -CA CAcert.pem -CAkey CAkey.pem -req -in
petic-cert-client.pem -set_serial 3 -days 15 -extfile config2.txt
-out client-cert.pem
```

1.4 Exportando los certificados de los clientes

Se exporta a un formato estándar que el navegador entiende: pkcs12.
Este paso se dará en el lado del cliente.
Con el certificado del cliente, la clave privada del cliente y el certificado de la autoridad certificadora se genera un fichero que es el que el cliente debe instalar en su navegador.



1.5 Definiendo la lista de revocación

El archivo que define los certificados revocados se encuentra en /etc/ssl/index.txt
Se configura correctamente /etc/ssl/openssl.cnf

```
#nsRevocationUrl
#nsRenewalUrl
#nsCaPolicyUrl
#nsSslServerName

# This really needs to be in place for it to be a proxy certificate.
proxyCertInfo=critical,language:id-ppl-anyLanguage,pathlen:3,policy:foo

####################################################################
[ tsa ]

default_tsa = tsa_config1       # the default TSA section

[ tsa_config1 ]

# These are used by the TSA reply generation only.
dir             = ./demoCA              # TSA root directory
serial          = $dir/tsaserial        # The current serial number (mandatory)
crypto_device   = builtin               # OpenSSL engine to use for signing
signer_cert     = $dir/tsacert.pem      # The TSA signing certificate
                                        # (optional)
certs           = $dir/CAcert.pem       # Certificate chain to include in reply
                                        # (optional)
signer_key      = $dir/private/tsakey.pem # The TSA private key (optional)
signer_digest  = sha256                 # Signing digest to use. (Optional)
default_policy  = tsa_policy1           # Policy if request did not specify it
                                        # (optional)
other_policies  = tsa_policy2, tsa_policy3      # acceptable policies (optional)
digests     = sha1, sha256, sha384, sha512  # Acceptable message digests (mandator
y)
accuracy        = secs:1, millisecs:500, microsecs:100  # (optional)
clock_precision_digits  = 0     # number of digits after dot. (optional)
ordering                = yes   # Is ordering defined for timestamps?
                                # (optional, default: no)
tsa_name                = yes   # Must the TSA name be included in the reply?
                                # (optional, default: no)
ess_cert_id_chain       = no    # Must the ESS cert id chain be included?
                                # (optional, default: no)
"openssl.cnf" 346L, 10774C                                    338,8       Final
```

En este fichero de configuración se ha especificado que el directorio donde van a estar el certificado, la clave privada y el número de serie de la CA es /etc/ssl/demoCA, la estructura de este directorio es la siguiente

```
root@turbomachine:/etc/ssl# tree demoCA/
demoCA/
├── CAcert.pem
├── CAcert.srl
├── index.txt
├── index.txt.attr
└── private
    └── CAkey.pem

1 directory, 5 files
```

Se crea un fichero crl y pem que representa la lista de revocación mediante el comando

```
openssl ca -gencrl -out listarev.pem
```

```
openssl ca -gencrl -out listarev.crl
```

```
root@turbomachine:/etc/ssl# cat listarev.crl
-----BEGIN X509 CRL-----
MIIB0jCBuzANBgkqhkiG9w0BAQsFADCBizELMAkGA1UEBhMCRVMxETAPBgNVBAgM
CFRlbmVyaWZlMRIwEAYDVQQHDAlMYSBMYWd1bmExDDAKBgNVBAoMA1VMTDENMAsG
A1UECwwERHB0bzEPMA0GA1UEAwwGTWFydGluMScwJQYJKoZIhvcNAQkBFhhhbHUw
MTAwODMyMjExQHVsbC5lZHUuZXMXDTE4MTEwOTEzNDA0MFoXDTE4MTIwOTEzNDA0
MFowDQYJKoZIhvcNAQELBQADggEBAJ6wuV+h9lFKAZm72UAhmJ8kSj1TQFKIMg+g
ELpl80E187OvFM9mr4sTS02cg01zly3a8bdATrUGLxB2k8+UP1ROGArhRWD0VmtY
FgImkDYZRHjLqkyi9LUVu2h8ns983RsojofWs/YnOXnRY/1qnWTOrdPEPGNv3/nn
tO2uha6gvzDEs1Dh+9LNx8rh7/MWTgUhSqA5CDI5fI1HGcRc3yup7KRvPLmoPxNe
+s8Gwv/+yeb8RW5PLPQU9K0VXPKKsPzVI58rz4PTJSqjm9c8MbME1QR3J3XHc8ER
us0isGrr02aOCVAl90Ohz/Nkji6xS+zjyJZwqX9wPK9M8JH4hrY=
-----END X509 CRL-----
```