Seguridad y protección de sistemas informáticos

Práctica 4: Certificados

Antes que nada como vamos a trabajar sobre un directorio root, hago `sudo -i` y enconces es cuando comienzo.

Para empezar, creo el directorio “ca” en root usando `mkdir /root/ca` . Dentro de **ca** creo los directorios certs, crl, newcerts y private. Creo también un index.html para llevar control sobre los certificados firmados.

```
# cd /root/ca
# mkdir certs crl newcerts private
# chmod 700 private
# touch index.txt
# echo 1000 > serial
```

He buscado en internet un archivo de configuración lo más completo posible para openssl. Este archivo (openssl.cnf) debe estar en **ca**. El archivo es el siguiente:

```
# OpenSSL root CA configuration file.
# Copy to `/root/ca/openssl.cnf`.

[ ca ]
# `man ca`
default_ca = CA_default

[ CA_default ]
# Directory and file locations.
dir               = /root/ca
certs             = $dir/certs
crl_dir           = $dir/crl
new_certs_dir     = $dir/newcerts
database          = $dir/index.txt
serial            = $dir/serial
RANDFILE          = $dir/private/.rand

# The root key and root certificate.
private_key       = $dir/private/ca.key.pem
certificate       = $dir/certs/ca.cert.pem

# For certificate revocation lists.
crlnumber         = $dir/crlnumber
crl               = $dir/crl/ca.crl.pem
crl_extensions    = crl_ext
default_crl_days  = 30

# SHA-1 is deprecated, so use SHA-2 instead.
default_md        = sha256

name_opt          = ca_default
cert_opt          = ca_default
default_days      = 375
preserve          = no
policy            = policy_strict

[ policy_strict ]
# The root CA should only sign intermediate certificates that match.
# See the POLICY FORMAT section of `man ca`.
countryName             = match
stateOrProvinceName     = match
organizationName        = match
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

[ policy_loose ]
# Allow the intermediate CA to sign a more diverse range of certificates.
# See the POLICY FORMAT section of the `ca` man page.
countryName             = optional
stateOrProvinceName     = optional
localityName            = optional
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

[ req ]
# Options for the `req` tool (`man req`).
default_bits        = 2048
distinguished_name  = req_distinguished_name
string_mask         = utf8only

# SHA-1 is deprecated, so use SHA-2 instead.
```

```
default_md          = sha256

# Extension to add when the -x509 option is used.
x509_extensions     = v3_ca

[ req_distinguished_name ]
# See <https://en.wikipedia.org/wiki/Certificate_signing_request>.
countryName                     = Country Name (2 letter code)
stateOrProvinceName             = State or Province Name
localityName                    = Locality Name
0.organizationName              = Organization Name
organizationalUnitName          = Organizational Unit Name
commonName                      = Common Name
emailAddress                    = Email Address

# Optionally, specify some defaults.
countryName_default             = GB
stateOrProvinceName_default     = England
localityName_default            =
0.organizationName_default      = Alice Ltd
organizationalUnitName_default  =
emailAddress_default            =

[ v3_ca ]
# Extensions for a typical CA (`man x509v3_config`).
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
basicConstraints = critical, CA:true
keyUsage = critical, digitalSignature, cRLSign, keyCertSign

[ v3_intermediate_ca ]
# Extensions for a typical intermediate CA (`man x509v3_config`).
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
basicConstraints = critical, CA:true, pathlen:0
keyUsage = critical, digitalSignature, cRLSign, keyCertSign

[ usr_cert ]
# Extensions for client certificates (`man x509v3_config`).
basicConstraints = CA:FALSE
nsCertType = client, email
nsComment = "OpenSSL Generated Client Certificate"
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer
keyUsage = critical, nonRepudiation, digitalSignature, keyEncipherment
extendedKeyUsage = clientAuth, emailProtection

[ server_cert ]
# Extensions for server certificates (`man x509v3_config`).
basicConstraints = CA:FALSE
nsCertType = server
nsComment = "OpenSSL Generated Server Certificate"
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer:always
keyUsage = critical, digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth

[ crl_ext ]
# Extension for CRLs (`man x509v3_config`).
authorityKeyIdentifier=keyid:always

[ ocsp ]
# Extension for OCSP signing certificates (`man ocsp`).
basicConstraints = CA:FALSE
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer
keyUsage = critical, digitalSignature
extendedKeyUsage = critical, OCSPSigning
```

En la parte de [ca], que es obligatoria, le decimos a openssl que use las opciones por defecto [CA_default] que posteriormente definimos. En [CA_default] indicamos los directorios que previamente hemos creado y definido. En [policy_strict] sirve para firmar certificados intermediariosn de root y [policy_loose] para todos.

[req] es para cuando creamos certificados o para peticiones de firma de certificados. (lo usaremos más tarde). También definimos [v3_ca] para aplicar detalles personalizados, en este caso lo usaremos cuando creemos un certificado root. [usr_cert] es una extensión para certificaciones de cliente, como los de usuario remoto. Del mismo modo usamos [server_cert] para certificaciones de servidor. Finalmente tenemos [crl_ext] para creater CRLs y [ocsp] para **Online Cerificate Status Protocol**.

Una vez definido el archivo, pasamos a crear la clave, como en las prácticas indica que no se use RSA, uso DSA, y para eso indico los parámetros, previamente.

```
openssl dsaparam -out dsaparam.pem 4096

Generating DSA parameters, 4096 bit long prime
This could take some time
```

Y una vez definidos los parámetros, ya si, creo la clave con:

```
openssl gendsa -aes256 -out private/ca.key.pem dsaparam.pem
Generating DSA key, 4096 bits
```

Me pide que cree una contraseña para la cave y uso `S0s4yweall`. Tras esto, uso openssl con el `req` que habíamos definido antes y la extensión `v3_ca` sando sha256, para crear el certificado.

```
openssl req -config openssl.cnf -key private/ca.key.pem -new -x509 -days 7300 -sha256 -extensions v3_ca -out certs/ca.cert.pem
```

Finalmente uso `chmod 400 private/ca.key.pem` Para dar permisos de lectura al dueño del certificado. Con `openssl x509 -noout -text -in certs/ca.cert.pem` podemos verificar nuestro certificado. En mi caso, el output es:

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            a0:8d:b3:34:50:39:1e:9b
    Signature Algorithm: dsa_with_SHA256
        Issuer: C=ES, ST=Spain, O=Paula co, CN=paula student
        Validity
            Not Before: Dec 11 10:46:21 2017 GMT
            Not After : Dec  6 10:46:21 2037 GMT
        Subject: C=ES, ST=Spain, O=Paula co, CN=paula student
        Subject Public Key Info:
            Public Key Algorithm: dsaEncryption
                pub:
                    40:59:25:28:f2:dd:7f:aa:a2:0a:88:51:13:cf:a3:
                    23:98:9b:88:65:83:5f:0e:2a:61:7f:84:f3:cb:a7:
                    72:ee:af:e1:c6:93:f2:ec:94:92:64:1e:33:c6:c0:
                    b5:20:ce:47:f6:12:d5:af:05:ef:30:d5:b1:c3:d4:
                    c5:62:50:c5:cf:0f:45:a3:fb:f6:ca:ad:96:f2:68:
                    07:ad:0b:78:94:bf:11:eb:b2:95:2e:b2:f8:22:21:
                    b4:33:76:25:bc:d8:6c:76:0f:73:a6:e7:76:5c:39:
                    27:2d:1a:91:ac:b0:26:03:c3:7c:18:f3:1e:70:45:
                    b5:b8:04:c6:b4:77:1b:19:60:95:8a:ed:f9:7d:18:
                    5d:72:2c:3b:73:12:de:dd:a3:13:21:33:50:8e:c0:
                    a1:b5:bd:50:00:ee:7b:48:5b:0b:c4:e0:02:8f:6a:
                    51:77:42:74:0a:08:0c:94:7d:f7:e8:b4:be:dc:5f:
                    a6:01:cd:ca:b6:18:74:00:c0:e8:04:8b:5c:1f:1a:
                    2f:0f:18:3d:1c:58:be:a5:01:f6:2a:ee:b8:c2:dd:
                    71:32:c4:b5:14:50:cd:50:79:db:53:b4:d9:45:83:
                    71:92:a7:c7:8e:71:45:4e:ca:e9:f4:81:a5:d6:a6:
                    a2:a0:ab:72:65:16:e0:e7:6f:22:60:47:ee:fc:1d:
                    0e:1a:3b:13:5b:43:c9:50:2e:4d:db:cf:ed:85:7f:
                    fc:b0:65:37:92:b6:4f:34:ff:64:e8:7a:ea:de:10:
                    cb:43:a6:85:eb:d4:af:dc:0a:16:3a:c4:14:a4:e4:
                    43:a8:2a:7f:6b:87:b4:2e:05:96:e5:a1:ce:db:7f:
                    0c:08:76:41:85:4f:55:01:93:e2:37:82:84:d7:34:
                    14:80:dd:b9:2d:d2:f2:6d:ae:0e:fd:61:fb:a7:c5:
                    08:d2:2f:27:c1:2d:61:89:8d:37:55:84:d1:00:a6:
                    21:b1:8c:2c:26:a5:93:06:83:71:e6:a4:05:ae:f5:
                    24:70:fc:75:0e:86:70:1a:2c:31:2f:d5:00:6d:95:
                    db:66:7f:4c:f4:e1:48:8b:5a:a9:12:54:5e:a2:c6:
                    3b:16:ea:d8:60:04:77:6f:80:f1:30:17:0a:bc:ba:
                    71:44:b1:19:53:ca:01:4c:af:05:d8:26:4b:f2:eb:
                    75:29:44:5d:a0:93:44:22:ea:29:6a:cd:71:da:a1:
                    ab:3a:c2:57:e6:8b:aa:dd:25:91:ff:78:58:b8:cc:
                    b9:7f:c5:44:0f:bd:ff:e2:7d:28:5e:f0:89:49:3b:
                    62:d4:84:a6:ce:14:c3:0c:42:81:17:de:d0:1c:a2:
                    6f:18:c8:d5:26:07:c4:64:a3:e3:35:93:55:d5:b5:
                    37:28
                P:
                    00:80:af:d2:8c:53:99:32:43:4f:a5:ed:4b:cf:35:
                    7c:b8:97:97:de:0e:e6:7e:4d:71:dc:d4:8f:ef:47:
                    61:bc:4b:47:c5:4e:a8:52:8b:12:94:af:9b:53:84:
                    c0:a6:7e:88:8b:fb:62:6f:77:63:cc:fc:f8:68:f6:
                    aa:eb:85:e1:33:09:70:41:15:66:e5:6d:0f:5d:68:
                    28:9c:1d:d2:18:bd:50:cc:d3:d8:19:9e:2d:05:e1:
                    cd:90:35:e7:27:f0:fc:c2:db:87:88:f2:0a:73:f1:
                    cb:11:bf:f1:fa:b6:0d:0b:bf:b3:57:35:09:36:2b:
                    f8:d3:e8:b2:80:aa:12:b7:9a:cf:4f:e6:01:1a:3c:
                    20:af:d4:bd:ab:fc:75:dc:8c:2e:f9:59:a1:50:22:
                    e4:67:3d:a2:37:0c:1f:ab:20:b4:77:01:55:17:3d:
                    16:d1:ac:87:e9:a8:44:39:59:19:43:ff:9a:99:9b:
                    71:84:e9:52:2d:b0:aa:e3:80:62:b2:b6:d5:f6:66:
                    1c:03:a2:51:28:7f:3b:9e:cb:85:da:b8:32:bf:68:
                    fa:8f:90:a1:6f:3a:aa:9e:0b:32:95:dd:46:09:58:
                    8f:13:57:70:36:52:67:2f:5d:0b:d7:e5:be:72:b0:
                    c8:f3:ac:97:44:a5:c1:10:9f:08:49:02:a1:77:48:
                    85:5c:72:55:b9:70:3a:09:af:1d:55:46:f6:44:be:
                    b0:29:42:ad:0c:71:66:40:57:91:48:aa:7b:53:0f:
                    f4:e7:c3:ac:28:2b:0b:12:9c:16:6b:61:ab:f2:6c:
```

```
                    c8:93:cd:ef:31:2e:4c:48:9a:a5:03:f1:7e:37:f7:
                    d9:0c:56:f9:3e:44:56:b4:0a:0d:ab:5b:5d:4d:bf:
                    9a:f2:33:b1:71:e9:71:b8:7b:d0:1a:07:7e:0e:24:
                    3e:a6:aa:87:01:6e:01:c7:44:5e:f8:99:ed:7a:e4:
                    08:25:df:3d:39:ba:1f:7a:1c:bd:9b:ec:0a:b6:2c:
                    a0:34:4d:b2:7b:77:3b:14:47:d6:29:5b:05:a4:71:
                    2c:82:ba:2a:2a:25:fc:57:18:70:ae:e9:af:b9:64:
                    05:98:c4:31:34:cb:94:aa:d9:11:86:a1:a8:6a:f5:
                    6e:3b:33:d8:a3:2b:58:d8:48:80:0b:ee:40:77:d4:
                    d5:15:d4:37:67:53:fd:e1:bf:89:c0:63:ad:25:80:
                    48:7e:09:2f:f2:a7:79:a3:4a:b4:77:b3:0a:f5:4d:
                    21:d9:80:49:21:a7:9c:f9:e1:71:8f:a7:59:14:cd:
                    da:0a:44:01:b1:1d:00:85:4d:94:62:36:a4:7e:85:
                    ef:1c:89:e1:da:28:df:f3:98:5d:41:b1:9e:d1:7d:
                    c6:59:85
                Q:
                    00:b7:b8:52:26:3e:10:77:d9:b4:0f:22:6f:6d:8e:
                    23:88:fd:60:64:15:89:91:8c:f5:00:ad:0b:2e:eb:
                    64:40:67
                G:
                    00:80:45:33:da:f2:57:b2:bd:42:b9:6e:85:b8:19:
                    82:e6:4f:a3:7f:da:6b:ef:54:f6:41:f6:ba:39:70:
                    cb:68:5d:07:ee:d1:a3:6b:b1:6a:d2:ca:a7:51:fb:
                    d5:45:64:ad:99:cd:e5:98:2f:e5:8c:b4:72:f8:28:
                    3c:6f:c5:2f:3a:de:40:77:66:17:30:38:3b:48:88:
                    1a:85:fb:04:59:48:2f:78:ce:46:ce:88:c3:f9:e0:
                    5c:d6:ab:c5:b5:ea:28:a9:c2:73:00:df:19:f2:b4:
                    e2:f0:cc:7a:bd:d9:13:88:d0:4f:47:74:15:10:7d:
                    ee:3f:6b:e4:86:f6:c0:63:fb:b0:cc:2b:cb:a7:a8:
                    0c:55:b6:9b:ad:e2:d1:a7:a6:67:16:4c:2e:a9:52:
                    00:52:49:7e:27:bc:9f:85:a5:f5:3b:25:23:d2:54:
                    ae:4b:18:39:54:36:cb:65:bc:ec:10:6d:2f:1b:9b:
                    90:03:e3:bf:80:ad:1d:db:0e:21:b8:0b:00:4f:04:
                    fe:0a:c4:91:ef:a4:f7:fe:eb:2c:64:f8:47:d0:33:
                    78:d3:4b:25:fa:23:c8:21:0f:41:89:5d:f5:fe:7c:
                    57:84:61:65:71:db:60:f5:9f:14:c0:3e:09:14:f5:
                    d0:76:4b:e3:bc:0a:9d:b8:ae:10:91:13:2c:e8:c6:
                    a4:c7:30:c3:2c:a3:07:8b:b7:91:8d:af:95:ec:18:
                    98:7c:6b:d8:9a:9c:e1:56:7b:74:1d:ab:af:d2:f4:
                    b5:b1:e1:80:32:74:ba:c6:30:7f:e4:17:08:02:3d:
                    e4:96:91:c0:9c:5c:be:8e:71:da:fa:8b:59:7f:bb:
                    c2:f4:f9:b2:29:23:fe:bb:7e:38:12:c0:00:34:78:
                    66:ef:e7:f0:53:6a:0b:fb:0d:ab:78:fa:d7:8d:45:
                    3c:60:82:14:bc:42:0f:bc:5d:e5:8f:af:be:30:66:
                    b5:f6:6f:fc:08:1d:31:20:2d:f5:4c:1a:b5:6e:7c:
                    74:39:47:47:08:8e:d6:e5:fd:e4:e7:6c:8d:ed:ba:
                    0a:12:a3:43:18:60:c1:07:c8:95:d2:92:bb:5c:a8:
                    98:7b:c8:ae:6b:88:3f:fb:0a:df:cf:a3:b2:6c:e2:
                    02:da:c4:43:84:8d:57:3a:ac:63:29:6e:52:b2:2c:
                    ea:66:a8:ed:ca:40:72:ec:43:90:92:54:0a:7c:ed:
                    b3:ce:67:7d:29:f7:0c:34:e6:5e:3b:f4:89:f5:bd:
                    45:30:75:1c:a0:1d:7e:7b:b7:bd:01:7b:cd:f4:3d:
                    21:b8:7f:97:86:9b:be:2d:1a:7b:89:d5:4c:12:8f:
                    1e:63:64:7e:d2:b9:eb:da:dc:52:47:a5:62:96:9c:
                    3b:81:d7
            X509v3 extensions:
                X509v3 Subject Key Identifier:
                    D2:66:DD:51:01:7F:D5:11:CC:D4:6A:8A:9B:DE:78:9E:A8:E5:55:62
                X509v3 Authority Key Identifier:
                    keyid:D2:66:DD:51:01:7F:D5:11:CC:D4:6A:8A:9B:DE:78:9E:A8:E5:55:62

                X509v3 Basic Constraints: critical
                    CA:TRUE
                X509v3 Key Usage: critical
                    Digital Signature, Certificate Sign, CRL Sign
        Signature Algorithm: dsa_with_SHA256
             r:
                 00:8e:61:33:84:d0:97:d5:07:f1:f7:0c:30:3a:37:
                 8b:32:b8:24:11:6d:75:d5:d5:a1:ce:91:2b:44:64:
                 25:70:98
             s:
                 7e:8a:74:51:7c:09:b2:ce:41:2e:a3:21:87:89:ab:
                 ff:b9:ff:7f:2f:18:86:ee:6a:2c:66:dd:30:06:3f:
                 be:5f
```

Para mayor seguridad, crearemos un certificado Intermediario. Este nos permitirá firmar en nombre de *root CA*, ahora mismo creado. Para ello, haremos un directorio intermediate dentro de `/root/ca` con `mkdir /root/ca/intermediate` y dentro crearemos los directorios certs, crl, csr, newcerts y private usando `mkdir certs crl csr newcerts private` y daremos permisos sobre private con `chmod 700 private`. Por último, como habíamos hecho antes creamos un index `touch index.txt` y hacemos `echo 1000 > serial`. Añadimos crlnumber para llevar un seguimiento de CRL con `echo 1000 > /root/ca/intermediate/crlnumber`. Usaremos un archivo de configuración parecido al anterior pero que en [CA_default] tenga `dir = /root/ca/intermediate` y los cambios relacionados. Para generar la clave intermediaria usaremos `gendsa` como antes pero **atención**, en `/root/ca`, un directorio superior a intermediate, y la clave es `intermediate.key.pem`. Para crear una petición de firma (CSR) intermediario, por tanto, usaremos:

```
openssl req -config intermediate/openssl.cnf -new -sha256 -key intermediate/private/intermediate.key.pem -out intermediate/csr/intermediate.csr.p
em
```

Y en '*Common Name*' deberemos poner que es intermediario. Para generar el certificado, usaremos (con la extensión v3):

```
openssl ca -config openssl.cnf -extensions v3_intermediate_ca -days 3650 -notext -md sha256 -in intermediate/csr/intermediate.csr.pem -out interm
ediate/certs/intermediate.cert.pem
```

Y añadimos los permisos con `chmod 444 intermediate/certs/intermediate.cert.pem`. Cuando definimos antes *Common name* hay que indicar el dominio (www.ejemplo.net) que queramos. Podemos verificar como antes usando `openssl x509 -noout -text -in intermediate/certs/intermediate.cert.pem` y verificar el certificado con `openssl verify -CAfile certs/ca.cert.pem intermediate/certs/intermediate.cert.pem` lo que, si está correcto, debería devolvernos un 'OK'. Finalmente crearemos un archivo cadena, que permite a la aplicación que comprueba nuestro certificado verificar el intermediario con el root. Para ello usamos `cat intermediate/certs/intermediate.cert.pem certs/ca.cert.pem > intermediate/certs/ca-chain.certs.pem` y damos permisos con `chmod 444 intermediate/certs/ca-chain.cert.pem`.