【转】OpenSSL command line Root and Intermediate CA including OCSP, CRL and revocation

2017-08-22

原文：
https://raymii.org/s/tutorials/OpenSSL_command_line_Root_and_Intermediate_CA_including_OCSP_CRL%20and_revocation.html

These are quick and dirty notes on generating a certificate authority (CA), intermediate certificate authorities and end certificates using OpenSSL. It includes OCSP, CRL and CA Issuer information and specific issue and expiry dates.

We'll set up our own root CA. We'll use the root CA to generate an example intermediate CA. We'll use the intermediate CA to sign end user certificates.

Root CA

Create and move in to a folder for the root ca:

```
mkdir ~/SSLCA/root/
cd ~/SSLCA/root/
```

Generate a 8192-bit long SHA-256 RSA key for our root CA:

```
openssl genrsa -aes256 -out rootca.key 8192
```

Example output:

```
Generating RSA private key, 8192 bit long modulus
.........++
..............................................................................................................................++
e is 65537 (0x10001)
```

If you want to password-protect this key, add the option `-aes256`.

Create the self-signed root CA certificate `ca.crt`; you'll need to provide an identity for your root CA:

```
openssl req -sha256 -new -x509 -days 1826 -key rootca.key -out rootca.crt
```

Example output:

```
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) [AU]:NL
State or Province Name (full name) [Some-State]:Zuid Holland
Locality Name (eg, city) []:Rotterdam
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Sparkling Network
Organizational Unit Name (eg, section) []:Sparkling CA
Common Name (e.g. server FQDN or YOUR name) []:Sparkling Root CA
Email Address []:
```

Create a few files where the CA will store it's serials:

```
touch certindex
echo 1000 > certserial
echo 1000 > crlnumber
```

Place the CA config file. This file has stubs for CRL and OCSP endpoints.

```
# vim ca.conf
[ ca ]
default_ca = myca

[ crl_ext ]
issuerAltName=issuer:copy
authorityKeyIdentifier=keyid:always

 [ myca ]
 dir = ./
 new_certs_dir = $dir
 unique_subject = no
 certificate = $dir/rootca.crt
 database = $dir/certindex
```

```
private_key = $dir/rootca.key
serial = $dir/certserial
default_days = 730
default_md = sha1
policy = myca_policy
x509_extensions = myca_extensions
crlnumber = $dir/crlnumber
default_crl_days = 730

[ myca_policy ]
commonName = supplied
stateOrProvinceName = supplied
countryName = optional
emailAddress = optional
organizationName = supplied
organizationalUnitName = optional

[ myca_extensions ]
basicConstraints = critical,CA:TRUE
keyUsage = critical,any
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
keyUsage = digitalSignature,keyEncipherment,cRLSign,keyCertSign
extendedKeyUsage = serverAuth
crlDistributionPoints = @crl_section
subjectAltName  = @alt_names
authorityInfoAccess = @ocsp_section

[ v3_ca ]
basicConstraints = critical,CA:TRUE,pathlen:0
keyUsage = critical,any
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
```

```
keyUsage = digitalSignature,keyEncipherment,cRLSign,keyCertSign
extendedKeyUsage = serverAuth
crlDistributionPoints = @crl_section
subjectAltName  = @alt_names
authorityInfoAccess = @ocsp_section

[alt_names]
DNS.0 = Sparkling Intermidiate CA 1
DNS.1 = Sparkling CA Intermidiate 1

[crl_section]
URI.0 = http://pki.sparklingca.com/SparklingRoot.crl
URI.1 = http://pki.backup.com/SparklingRoot.crl

[ocsp_section]
caIssuers;URI.0 = http://pki.sparklingca.com/SparklingRoot.crt
caIssuers;URI.1 = http://pki.backup.com/SparklingRoot.crt
OCSP;URI.0 = http://pki.sparklingca.com/ocsp/
OCSP;URI.1 = http://pki.backup.com/ocsp/
```

If you need to set a specific certificate start / expiry date, add the following to `[myca]`

```
# format: YYYYMMDDHHMMSS
default_enddate = 20191222035911
default_startdate = 20181222035911
```

Creating Intermediate 1 CA

Generate the intermediate CA's private key:

```
openssl genrsa -out intermediate1.key 4096
```

Generate the intermediate1 CA's CSR:

```
openssl req -new -sha256 -key intermediate1.key -out intermediate1.csr
```

Example output:

```
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) [AU]:NL
State or Province Name (full name) [Some-State]:Zuid Holland
Locality Name (eg, city) []:Rotterdam
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Sparkling Network
Organizational Unit Name (eg, section) []:Sparkling CA
Common Name (e.g. server FQDN or YOUR name) []:Sparkling Intermediate CA
Email Address []:

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

Make sure the subject (CN) of the intermediate is different from the root.

Sign the intermediate1 CSR with the Root CA:

```
openssl ca -batch -config ca.conf -notext -in intermediate1.csr -out intermediat
e1.crt
```

Example Output:

```
Using configuration from ca.conf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'NL'
stateOrProvinceName    :ASN.1 12:'Zuid Holland'
localityName          :ASN.1 12:'Rotterdam'
```

```
organizationName      :ASN.1 12:'Sparkling Network'
organizationalUnitName:ASN.1 12:'Sparkling CA'
commonName            :ASN.1 12:'Sparkling Intermediate CA'
Certificate is to be certified until Mar 30 15:07:43 2017 GMT (730 days)


Write out database with 1 new entries
Data Base Updated
```

Generate the CRL (both in PEM and DER):

```
openssl ca -config ca.conf -gencrl -keyfile rootca.key -cert rootca.crt -out roo
tca.crl.pem


openssl crl -inform PEM -in rootca.crl.pem -outform DER -out rootca.crl
```

Generate the CRL after every certificate you sign with the CA.

If you ever need to revoke the this intermediate cert:

```
openssl ca -config ca.conf -revoke intermediate1.crt -keyfile rootca.key -cert r
ootca.crt
```

Configuring the Intermediate CA 1

Create a new folder for this intermediate and move in to it:

```
mkdir ~/SSLCA/intermediate1/
cd ~/SSLCA/intermediate1/
```

Copy the Intermediate cert and key from the Root CA:

```
cp ~/SSLCA/root/intermediate1.key ./
cp ~/SSLCA/root/intermediate1.crt ./
```

Create the index files:

```
touch certindex
echo 1000 > certserial
echo 1000 > crlnumber
```

Create a new `ca.conf` file:

```
# vim ca.conf
[ ca ]
default_ca = myca

[ crl_ext ]
issuerAltName=issuer:copy
authorityKeyIdentifier=keyid:always

 [ myca ]
 dir = ./
 new_certs_dir = $dir
 unique_subject = no
 certificate = $dir/intermediate1.crt
 database = $dir/certindex
 private_key = $dir/intermediate1.key
 serial = $dir/certserial
 default_days = 365
 default_md = sha1
 policy = myca_policy
 x509_extensions = myca_extensions
 crlnumber = $dir/crlnumber
 default_crl_days = 365

 [ myca_policy ]
 commonName = supplied
 stateOrProvinceName = supplied
 countryName = optional
 emailAddress = optional
 organizationName = supplied
 organizationalUnitName = optional

 [ myca_extensions ]
```

```
basicConstraints = critical,CA:FALSE
keyUsage = critical,any
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
keyUsage = digitalSignature,keyEncipherment
extendedKeyUsage = serverAuth
crlDistributionPoints = @crl_section
subjectAltName  = @alt_names
authorityInfoAccess = @ocsp_section

[alt_names]
DNS.0 = example.com
DNS.1 = example.org

[crl_section]
URI.0 = http://pki.sparklingca.com/SparklingIntermidiate1.crl
URI.1 = http://pki.backup.com/SparklingIntermidiate1.crl

[ocsp_section]
caIssuers;URI.0 = http://pki.sparklingca.com/SparklingIntermediate1.crt
caIssuers;URI.1 = http://pki.backup.com/SparklingIntermediate1.crt
OCSP;URI.0 = http://pki.sparklingca.com/ocsp/
OCSP;URI.1 = http://pki.backup.com/ocsp/
```

Change the `[alt_names]` section to whatever you need as Subject Alternative names. Remove it including the `subjectAltName  = @alt_names` line if you don't want a Subject Alternative Name.

If you need to set a specific certificate start / expiry date, add the following to `[myca]`

```
# format: YYYYMMDDHHMMSS
default_enddate = 20191222035911
default_startdate = 20181222035911
```

Generate an empty CRL (both in PEM and DER):

```
openssl ca -config ca.conf -gencrl -keyfile rootca.key -cert rootca.crt -out roo
tca.crl.pem
```

```
openssl crl -inform PEM -in rootca.crl.pem -outform DER -out rootca.crl
```

Creating end user certificates

We use this new intermediate CA to generate an end user certificate. Repeat these steps for every end user certificate you want to sign with this CA.

```
mkdir enduser-certs
```

Generate the end user's private key:

```
openssl genrsa -out enduser-certs/enduser-example.com.key 4096
```

Generate the end user's CSR:

```
openssl req -new -sha256 -key enduser-certs/enduser-example.com.key -out enduse
r-certs/enduser-example.com.csr
```

Example output:

```
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) [AU]:NL

State or Province Name (full name) [Some-State]:Noord Holland

Locality Name (eg, city) []:Amsterdam

Organization Name (eg, company) [Internet Widgits Pty Ltd]:Example Inc

Organizational Unit Name (eg, section) []:IT Dept

Common Name (e.g. server FQDN or YOUR name) []:example.com

Email Address []:


Please enter the following 'extra' attributes

to be sent with your certificate request
```

```
A challenge password []:

An optional company name []:
```

Sign the end user's CSR with the Intermediate 1 CA:

```
openssl ca -batch -config ca.conf -notext -in enduser-certs/enduser-example.co
m.csr -out enduser-certs/enduser-example.com.crt
```

Example output:

```
Using configuration from ca.conf

Check that the request matches the signature

Signature ok

The Subject's Distinguished Name is as follows

countryName           :PRINTABLE:'NL'

stateOrProvinceName    :ASN.1 12:'Noord Holland'

localityName          :ASN.1 12:'Amsterdam'

organizationName      :ASN.1 12:'Example Inc'

organizationalUnitName:ASN.1 12:'IT Dept'

commonName            :ASN.1 12:'example.com'

Certificate is to be certified until Mar 30 15:18:26 2016 GMT (365 days)


Write out database with 1 new entries

Data Base Updated
```

Generate the CRL (both in PEM and DER):

```
openssl ca -config ca.conf -gencrl -keyfile intermediate1.key -cert intermediate
1.crt -out intermediate1.crl.pem


openssl crl -inform PEM -in intermediate1.crl.pem -outform DER -out intermediate
1.crl
```

Generate the CRL after every certificate you sign with the CA.

If you ever need to revoke the this end users cert:

```
openssl ca -config ca.conf -revoke enduser-certs/enduser-example.com.crt -keyfi
le intermediate1.key -cert intermediate1.crt
```

Example output:

```
Using configuration from ca.conf

Revoking Certificate 1000.

Data Base Updated
```

Create the certificate chain file by concatenating the Root and intermediate 1 certificates together.

```
cat ../root/rootca.crt intermediate1.crt > enduser-certs/enduser-example.com.ch
ain
```

Send the following files to the end user:

```
enduser-example.com.crt

enduser-example.com.key

enduser-example.com.chain
```

You can also let the end user supply their own CSR and just send them the `.crt` file. Do not delete that from the server, otherwise you cannot revoke it.

Validating the certificate

You can validate the end user certificate against the chain using the following command:

```
openssl verify -CAfile enduser-certs/enduser-example.com.chain enduser-certs/en
duser-example.com.crt

enduser-certs/enduser-example.com.crt: OK
```

You can also validate it against the CRL. Concatenate the PEM CRL and the chain together first:

```
cat ../root/rootca.crt intermediate1.crt intermediate1.crl.pem > enduser-certs/
enduser-example.com.crl.chain
```

Verify the certificate:

```
openssl verify -crl_check -CAfile enduser-certs/enduser-example.com.crl.chain e
nduser-certs/enduser-example.com.crt
```

Output when not revoked:

```
enduser-certs/enduser-example.com.crt: OK
```

Output when revoked:

```
enduser-certs/enduser-example.com.crt: CN = example.com, ST = Noord Holland, C =
 NL, O = Example Inc, OU = IT Dept
```

```
error 23 at 0 depth lookup:certificate revoked
```