



*Full credit is given to the above companies including the OS that this PDF file was generated!*

## ***Red Hat Enterprise Linux Release 9.2 Manual Pages on 'openssl-ca.1ossil' command***

***\$ man openssl-ca.1ossil***

OPENSSL-CA(1ossil)            OpenSSL            OPENSSL-CA(1ossil)

### NAME

openssl-ca - sample minimal CA application

### SYNOPSIS

```
openssl ca [-help] [-verbose] [-config filename] [-name section]
[-section section] [-genctrl] [-revoke file] [-valid file] [-status
serial] [-updatedb] [-crl_reason reason] [-crl_hold instruction]
[-crl_compromise time] [-crl_CA_compromise time] [-crl_lastupdate date]
[-crl_nextupdate date] [-crl_days days] [-crl_hours hours] [-crlsec
seconds] [-crl_exts section] [-startdate date] [-enddate date] [-days
arg] [-md arg] [-policy arg] [-keyfile filename|uri] [-keyform
DER|PEM|P12|ENGINE] [-key arg] [-passin arg] [-cert file] [-certform
DER|PEM|P12] [-selfsign] [-in file] [-inform DER|<PEM>] [-out file]
[-notext] [-dateopt] [-outdir dir] [-infile] [-spkac file] [-ss_cert
file] [-preserveDN] [-noemailDN] [-batch] [-msie_hack] [-extensions
section] [-extfile section] [-subj arg] [-utf8] [-sigopt nm:v] [-vfyopt
nm:v] [-create_serial] [-rand_serial] [-multivalue-rdn] [-rand files]
[-writerand file] [-engine id] [-provider name] [-provider-path path]
[-propquery propq] [certreq...]
```

### DESCRIPTION

This command emulates a CA application. See the WARNINGS especially when considering to use it productively. It can be used to sign certificate requests (CSRs) in a variety of forms and generate

certificate revocation lists (CRLs). It also maintains a text database of issued certificates and their status. When signing certificates, a single request can be specified with the `-in` option, or multiple requests can be processed by specifying a set of `certreq` files after all options.

Note that there are also very lean ways of generating certificates: the `req` and `x509` commands can be used for directly creating certificates. See `openssl-req(1)` and `openssl-x509(1)` for details.

The descriptions of the `ca` command options are divided into each purpose.

## OPTIONS

`-help`

Print out a usage message.

`-verbose`

This prints extra details about the operations being performed.

`-config filename`

Specifies the configuration file to use. Optional; for a description of the default value, see "COMMAND SUMMARY" in `openssl(1)`.

`-name section, -section section`

Specifies the configuration file section to use (overrides `default_ca` in the `ca` section).

`-in filename`

An input filename containing a single certificate request (CSR) to be signed by the CA.

`-inform DER|PEM`

The format of the data in certificate request input files; unspecified by default. See `openssl-format-options(1)` for details.

`-ss_cert filename`

A single self-signed certificate to be signed by the CA.

`-spkac filename`

A file containing a single Netscape signed public key and challenge and additional field values to be signed by the CA. See the SPKAC

FORMAT section for information on the required input and output format.

-infile

If present this should be the last option, all subsequent arguments are taken as the names of files containing certificate requests.

-out filename

The output file to output certificates to. The default is standard output. The certificate details will also be printed out to this file in PEM format (except that -spkac outputs DER format).

-outdir directory

The directory to output certificates to. The certificate will be written to a filename consisting of the serial number in hex with .pem appended.

-cert filename

The CA certificate, which must match with -keyfile.

-certform DER|PEM|P12

The format of the data in certificate input files; unspecified by default. See openssl-format-options(1) for details.

-keyfile filename|uri

The CA private key to sign certificate requests with. This must match with -cert.

-keyform DER|PEM|P12|ENGINE

The format of the private key input file; unspecified by default. See openssl-format-options(1) for details.

-sigopt nm:v

Pass options to the signature algorithm during sign operations.

Names and values of these options are algorithm-specific.

-vfyopt nm:v

Pass options to the signature algorithm during verify operations.

Names and values of these options are algorithm-specific.

This often needs to be given while signing too, because the self-signature of a certificate signing request (CSR) is verified against the included public key, and that verification may need its

own set of options.

**-key password**

The password used to encrypt the private key. Since on some systems the command line arguments are visible (e.g., when using ps(1) on

Unix), this option should be used with caution. Better use

**-passin.**

**-passin arg**

The key password source for key files and certificate PKCS#12

files. For more information about the format of arg see

**openssl-passphrase-options(1).**

**-selfsign**

Indicates the issued certificates are to be signed with the key the certificate requests were signed with (given with **-keyfile**).

Certificate requests signed with a different key are ignored. If

**-spkac**, **-ss\_cert** or **-gencl** are given, **-selfsign** is ignored.

A consequence of using **-selfsign** is that the self-signed certificate appears among the entries in the certificate database (see the configuration option database), and uses the same serial number counter as all other certificates sign with the self-signed certificate.

**-notext**

Don't output the text form of a certificate to the output file.

**-dateopt**

Specify the date output format. Values are: **rfc\_822** and **iso\_8601**.

Defaults to **rfc\_822**.

**-startdate date**

This allows the start date to be explicitly set. The format of the date is **YYMMDDHHMMSSZ** (the same as an ASN1 UTCTime structure), or **YYYYMMDDHHMMSSZ** (the same as an ASN1 GeneralizedTime structure). In both formats, seconds **SS** and timezone **Z** must be present.

**-enddate date**

This allows the expiry date to be explicitly set. The format of the date is **YYMMDDHHMMSSZ** (the same as an ASN1 UTCTime structure), or

YYYYMMDDHHMMSSZ (the same as an ASN1 GeneralizedTime structure). In

both formats, seconds SS and timezone Z must be present.

-days arg

The number of days to certify the certificate for.

-md alg

The message digest to use. Any digest supported by the openssl-dgst(1) command can be used. For signing algorithms that do not support a digest (i.e. Ed25519 and Ed448) any message digest that is set is ignored. This option also applies to CRLs.

-policy arg

This option defines the CA "policy" to use. This is a section in the configuration file which decides which fields should be mandatory or match the CA certificate. Check out the POLICY FORMAT section for more information.

-msie\_hack

This is a deprecated option to make this command work with very old versions of the IE certificate enrollment control "certenr3". It used UniversalStrings for almost everything. Since the old control has various security bugs its use is strongly discouraged.

-preserveDN

Normally the DN order of a certificate is the same as the order of the fields in the relevant policy section. When this option is set the order is the same as the request. This is largely for compatibility with the older IE enrollment control which would only accept certificates if their DN's match the order of the request.

This is not needed for Xenroll.

-noemailDN

The DN of a certificate can contain the EMAIL field if present in the request DN, however, it is good policy just having the e-mail set into the altName extension of the certificate. When this option is set the EMAIL field is removed from the certificate's subject and set only in the, eventually present, extensions. The email\_in\_dn keyword can be used in the configuration file to enable this

behaviour.

#### -batch

This sets the batch mode. In this mode no questions will be asked and all certificates will be certified automatically.

#### -extensions section

The section of the configuration file containing certificate extensions to be added when a certificate is issued (defaults to x509\_extensions unless the -extfile option is used). If no X.509 extensions are specified then a V1 certificate is created, else a V3 certificate is created. See the x509v3\_config(5) manual page for details of the extension section format.

#### -extfile file

An additional configuration file to read certificate extensions from (using the default section unless the -extensions option is also used).

#### -subj arg

Supersedes subject name given in the request.

The arg must be formatted as

"/type0=value0/type1=value1/type2=...". Special characters may be escaped by "\" (backslash), whitespace is retained. Empty values are permitted, but the corresponding type will not be included in the resulting certificate. Giving a single "/" will lead to an empty sequence of RDNs (a NULL-DN). Multi-valued RDNs can be formed by placing a "+" character instead of a "/" between the AttributeValueAssertions (AVAs) that specify the members of the set. Example:

```
"/DC=org/DC=OpenSSL/DC=users/UID=123456+CN=John Doe"
```

#### -utf8

This option causes field values to be interpreted as UTF8 strings, by default they are interpreted as ASCII. This means that the field values, whether prompted from a terminal or obtained from a configuration file, must be valid UTF8 strings.

#### -create\_serial

If reading serial from the text file as specified in the configuration fails, specifying this option creates a new random serial to be used as next serial number. To get random serial numbers, use the `-rand_serial` flag instead; this should only be used for simple error-recovery.

`-rand_serial`

Generate a large random number to use as the serial number. This overrides any option or configuration to use a serial number file.

`-multivalue-rdn`

This option has been deprecated and has no effect.

`-rand files`, `-writerand file`

See "Random State Options" in `openssl(1)` for details.

`-engine id`

See "Engine Options" in `openssl(1)`. This option is deprecated.

`-provider name`

`-provider-path path`

`-propquery propq`

See "Provider Options" in `openssl(1)`, `provider(7)`, and `property(7)`.

## CRL OPTIONS

`-gencrl`

This option generates a CRL based on information in the index file.

`-crl_lastupdate time`

Allows the value of the CRL's `lastUpdate` field to be explicitly set; if this option is not present, the current time is used.

Accepts times in `YYMMDDHHMMSSZ` format (the same as an ASN1 `UTCTime` structure) or `YYYYMMDDHHMMSSZ` format (the same as an ASN1 `GeneralizedTime` structure).

`-crl_nextupdate time`

Allows the value of the CRL's `nextUpdate` field to be explicitly set; if this option is present, any values given for `-crl days`, `-crl hours` and `-crl sec` are ignored. Accepts times in the same formats as `-crl_lastupdate`.

`-crl days num`

The number of days before the next CRL is due. That is the days from now to place in the CRL nextUpdate field.

-crlhours num

The number of hours before the next CRL is due.

-crlsec num

The number of seconds before the next CRL is due.

-revoke filename

A filename containing a certificate to revoke.

-valid filename

A filename containing a certificate to add a Valid certificate entry.

-status serial

Displays the revocation status of the certificate with the specified serial number and exits.

-updatedb

Updates the database index to purge expired certificates.

-crl\_reason reason

Revocation reason, where reason is one of: unspecified, keyCompromise, CACompromise, affiliationChanged, superseded, cessationOfOperation, certificateHold or removeFromCRL. The matching of reason is case insensitive. Setting any revocation reason will make the CRL v2.

In practice removeFromCRL is not particularly useful because it is only used in delta CRLs which are not currently implemented.

-crl\_hold instruction

This sets the CRL revocation reason code to certificateHold and the hold instruction to instruction which must be an OID. Although any OID can be used only holdInstructionNone (the use of which is discouraged by RFC2459) holdInstructionCallIssuer or holdInstructionReject will normally be used.

-crl\_compromise time

This sets the revocation reason to keyCompromise and the compromise time to time. time should be in GeneralizedTime format that is

YYYYMMDDHHMMSSZ.

-crl\_CA\_compromise time

This is the same as crl\_compromise except the revocation reason is set to CACompromise.

-crl\_exts section

The section of the configuration file containing CRL extensions to include. If no CRL extension section is present then a V1 CRL is created, if the CRL extension section is present (even if it is empty) then a V2 CRL is created. The CRL extensions specified are CRL extensions and not CRL entry extensions. It should be noted that some software (for example Netscape) can't handle V2 CRLs. See x509v3\_config(5) manual page for details of the extension section format.

## CONFIGURATION FILE OPTIONS

The section of the configuration file containing options for this command is found as follows: If the -name command line option is used, then it names the section to be used. Otherwise the section to be used must be named in the default\_ca option of the ca section of the configuration file (or in the default section of the configuration file). Besides default\_ca, the following options are read directly from the ca section:

RANDFILE

preserve

msie\_hack With the exception of RANDFILE, this is probably a bug and may change in future releases.

Many of the configuration file options are identical to command line options. Where the option is present in the configuration file and the command line the command line value is used. Where an option is described as mandatory then it must be present in the configuration file or the command line equivalent (if any) used.

oid\_file

This specifies a file containing additional OBJECT IDENTIFIERS.

Each line of the file should consist of the numerical form of the

object identifier followed by whitespace then the short name followed by whitespace and finally the long name.

#### oid\_section

This specifies a section in the configuration file containing extra object identifiers. Each line should consist of the short name of the object identifier followed by = and the numerical form. The short and long names are the same when this option is used.

#### new\_certs\_dir

The same as the -outdir command line option. It specifies the directory where new certificates will be placed. Mandatory.

#### certificate

The same as -cert. It gives the file containing the CA certificate.

Mandatory.

#### private\_key

Same as the -keyfile option. The file containing the CA private key. Mandatory.

#### RANDFILE

At startup the specified file is loaded into the random number generator, and at exit 256 bytes will be written to it. (Note: Using a RANDFILE is not necessary anymore, see the "HISTORY" section.

#### default\_days

The same as the -days option. The number of days to certify a certificate for.

#### default\_startdate

The same as the -startdate option. The start date to certify a certificate for. If not set the current time is used.

#### default\_enddate

The same as the -enddate option. Either this option or default\_days (or the command line equivalents) must be present.

#### default\_crl\_hours default\_crl\_days

The same as the -crlhours and the -crldays options. These will only be used if neither command line option is present. At least one of

these must be present to generate a CRL.

#### default\_md

The same as the -md option. Mandatory except where the signing algorithm does not require a digest (i.e. Ed25519 and Ed448).

#### database

The text database file to use. Mandatory. This file must be present though initially it will be empty.

#### unique\_subject

If the value yes is given, the valid certificate entries in the database must have unique subjects. If the value no is given, several valid certificate entries may have the exact same subject. The default value is yes, to be compatible with older (pre 0.9.8) versions of OpenSSL. However, to make CA certificate roll-over easier, it's recommended to use the value no, especially if combined with the -selfsign command line option.

Note that it is valid in some circumstances for certificates to be created without any subject. In the case where there are multiple certificates without subjects this does not count as a duplicate.

#### serial

A text file containing the next serial number to use in hex.

Mandatory. This file must be present and contain a valid serial number.

#### crlnumber

A text file containing the next CRL number to use in hex. The crl number will be inserted in the CRLs only if this file exists. If this file is present, it must contain a valid CRL number.

#### x509\_extensions

A fallback to the -extensions option.

#### crl\_extensions

A fallback to the -crlxts option.

#### preserve

The same as -preserveDN

#### email\_in\_dn

The same as -noemailDN. If you want the EMAIL field to be removed from the DN of the certificate simply set this to 'no'. If not present the default is to allow for the EMAIL field in the certificate's DN.

#### msie\_hack

The same as -msie\_hack

#### policy

The same as -policy. Mandatory. See the POLICY FORMAT section for more information.

#### name\_opt, cert\_opt

These options allow the format used to display the certificate details when asking the user to confirm signing. All the options supported by the x509 utilities -nameopt and -certopt switches can be used here, except the no\_signame and no\_sigdump are permanently set and cannot be disabled (this is because the certificate signature cannot be displayed because the certificate has not been signed at this point).

For convenience the values ca\_default are accepted by both to produce a reasonable output.

If neither option is present the format used in earlier versions of OpenSSL is used. Use of the old format is strongly discouraged because it only displays fields mentioned in the policy section, mishandles multicharacter string types and does not display extensions.

#### copy\_extensions

Determines how extensions in certificate requests should be handled. If set to none or this option is not present then extensions are ignored and not copied to the certificate. If set to copy then any extensions present in the request that are not already present are copied to the certificate. If set to copyall then all extensions in the request are copied to the certificate: if the extension is already present in the certificate it is deleted first. See the WARNINGS section before using this option.

The main use of this option is to allow a certificate request to supply values for certain extensions such as subjectAltName.

## POLICY FORMAT

The policy section consists of a set of variables corresponding to certificate DN fields. If the value is "match" then the field value must match the same field in the CA certificate. If the value is "supplied" then it must be present. If the value is "optional" then it may be present. Any fields not mentioned in the policy section are silently deleted, unless the -preserveDN option is set but this can be regarded more of a quirk than intended behaviour.

## SPKAC FORMAT

The input to the -spkac command line option is a Netscape signed public key and challenge. This will usually come from the KEYGEN tag in an HTML form to create a new private key. It is however possible to create SPKACs using openssl-spkac(1).

The file should contain the variable SPKAC set to the value of the SPKAC and also the required DN components as name value pairs. If you need to include the same component twice then it can be preceded by a number and a '!'.

When processing SPKAC format, the output is DER if the -out flag is used, but PEM format if sending to stdout or the -outdir flag is used.

## EXAMPLES

Note: these examples assume that the directory structure this command assumes is already set up and the relevant files already exist. This usually involves creating a CA certificate and private key with openssl-req(1), a serial number file and an empty index file and placing them in the relevant directories.

To use the sample configuration file below the directories demoCA, demoCA/private and demoCA/newcerts would be created. The CA certificate would be copied to demoCA/cacert.pem and its private key to demoCA/private/cakey.pem. A file demoCA/serial would be created containing for example "01" and the empty index file demoCA/index.txt.

Sign a certificate request:

```
openssl ca -in req.pem -out newcert.pem
```

Sign an SM2 certificate request:

```
openssl ca -in sm2.csr -out sm2.crt -md sm3 \  
-sigopt "distid:1234567812345678" \  
-vfyopt "distid:1234567812345678"
```

Sign a certificate request, using CA extensions:

```
openssl ca -in req.pem -extensions v3_ca -out newcert.pem
```

Generate a CRL

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

Sign several requests:

```
openssl ca -infiles req1.pem req2.pem req3.pem
```

Certify a Netscape SPKAC:

```
openssl ca -spkac spkac.txt
```

A sample SPKAC file (the SPKAC line has been truncated for clarity):

```
SPKAC=MIG0MGAwXDANBqkqhkiG9w0BAQEFAANLADBIACEAn7PDhCeV/xIxUg8V70YRxEK2A5
```

```
CN=Steve Test
```

```
emailAddress=steve@openssl.org
```

```
0.OU=OpenSSL Group
```

```
1.OU=Another Group
```

A sample configuration file with the relevant sections for this

command:

```
[ ca ]  
default_ca = CA_default # The default ca section  
[ CA_default ]  
dir = ./demoCA # top dir  
database = $dir/index.txt # index file.  
new_certs_dir = $dir/newcerts # new certs dir  
certificate = $dir/cacert.pem # The CA cert  
serial = $dir/serial # serial no file  
#rand_serial = yes # for random serial#'s  
private_key = $dir/private/cakey.pem# CA private key  
default_days = 365 # how long to certify for  
default_crl_days= 30 # how long before next CRL
```

```
default_md    = md5          # md to use
policy       = policy_any   # default policy
email_in_dn  = no           # Don't add the email into cert DN
name_opt     = ca_default   # Subject name display option
cert_opt     = ca_default   # Certificate display option
copy_extensions = none      # Don't copy extensions from request
[ policy_any ]
countryName   = supplied
stateOrProvinceName = optional
organizationName = optional
organizationalUnitName = optional
commonName    = supplied
emailAddress  = optional
```

## FILES

Note: the location of all files can change either by compile time options, configuration file entries, environment variables or command line options. The values below reflect the default values.

```
/usr/local/ssl/lib/openssl.cnf - master configuration file
./demoCA                       - main CA directory
./demoCA/cacert.pem           - CA certificate
./demoCA/private/cakey.pem    - CA private key
./demoCA/serial               - CA serial number file
./demoCA/serial.old           - CA serial number backup file
./demoCA/index.txt            - CA text database file
./demoCA/index.txt.old        - CA text database backup file
./demoCA/certs                 - certificate output file
```

## RESTRICTIONS

The text database index file is a critical part of the process and if corrupted it can be difficult to fix. It is theoretically possible to rebuild the index file from all the issued certificates and a current CRL: however there is no option to do this.

V2 CRL features like delta CRLs are not currently supported.

Although several requests can be input and handled at once it is only

possible to include one SPKAC or self-signed certificate.

## BUGS

This command is quirky and at times downright unfriendly.

The use of an in-memory text database can cause problems when large numbers of certificates are present because, as the name implies the database has to be kept in memory.

This command really needs rewriting or the required functionality exposed at either a command or interface level so that a more user-friendly replacement could handle things properly. The script CA.pl helps a little but not very much.

Any fields in a request that are not present in a policy are silently deleted. This does not happen if the `-preserveDN` option is used. To enforce the absence of the EMAIL field within the DN, as suggested by RFCs, regardless the contents of the request' subject the `-noemailDN` option can be used. The behaviour should be more friendly and configurable.

Canceling some commands by refusing to certify a certificate can create an empty file.

## WARNINGS

This command was originally meant as an example of how to do things in a CA. Its code does not have production quality. It was not supposed to be used as a full blown CA itself, nevertheless some people are using it for this purpose at least internally. When doing so, specific care should be taken to properly secure the private key(s) used for signing certificates. It is advisable to keep them in a secure HW storage such as a smart card or HSM and access them via a suitable engine or crypto provider.

This command command is effectively a single user command: no locking is done on the various files and attempts to run more than one `openssl ca` command on the same database can have unpredictable results.

The `copy_extensions` option should be used with caution. If care is not taken then it can be a security risk. For example if a certificate request contains a `basicConstraints` extension with `CA:TRUE` and the

copy\_extensions value is set to copyall and the user does not spot this when the certificate is displayed then this will hand the requester a valid CA certificate. This situation can be avoided by setting copy\_extensions to copy and including basicConstraints with CA:FALSE in the configuration file. Then if the request contains a basicConstraints extension it will be ignored.

It is advisable to also include values for other extensions such as keyUsage to prevent a request supplying its own values.

Additional restrictions can be placed on the CA certificate itself.

For example if the CA certificate has:

```
basicConstraints = CA:TRUE, pathlen:0
```

then even if a certificate is issued with CA:TRUE it will not be valid.

## HISTORY

Since OpenSSL 1.1.1, the program follows RFC5280. Specifically, certificate validity period (specified by any of -startdate, -enddate and -days) and CRL last/next update time (specified by any of -crl\_lastupdate, -crl\_nextupdate, -crl days, -crl hours and -crl sec) will be encoded as UTCTime if the dates are earlier than year 2049 (included), and as GeneralizedTime if the dates are in year 2050 or later.

OpenSSL 1.1.1 introduced a new random generator (CSPRNG) with an improved seeding mechanism. The new seeding mechanism makes it unnecessary to define a RANDFILE for saving and restoring randomness.

This option is retained mainly for compatibility reasons.

The -section option was added in OpenSSL 3.0.0.

The -multivalue-rdn option has become obsolete in OpenSSL 3.0.0 and has no effect.

The -engine option was deprecated in OpenSSL 3.0.

## SEE ALSO

openssl(1), openssl-req(1), openssl-spkac(1), openssl-x509(1),  
CA.pl(1), config(5), x509v3\_config(5)

## COPYRIGHT

Copyright 2000-2021 The OpenSSL Project Authors. All Rights Reserved.

Licensed under the Apache License 2.0 (the "License"). You may not use this file except in compliance with the License. You can obtain a copy in the file LICENSE in the source distribution or at <https://www.openssl.org/source/license.html>.

3.0.7

2023-07-13

OPENSSL-CA(1ossl)