



Rocky Enterprise Linux 9.2 Manual Pages on command 'req.1ssl'

C:\>man req.1ssl

REQ(1SSL) OpenSSL REQ(1SSL)

NAME

openssl-req, req - PKCS#10 certificate request and certificate generating utility

SYNOPSIS

```
openssl req [-help] [-inform PEM|DER] [-outform PEM|DER] [-in filename] [-passin
arg] [-out filename] [-passout arg] [-text] [-pubkey] [-noout] [-verify] [-modulus]
[-new] [-rand file...] [-writerand file] [-newkey rsa:bits] [-newkey alg:file]
[-nodes] [-key filename] [-keyform PEM|DER] [-keyout filename] [-keygen_engine id]
[-digest] [-config filename] [-multivalue-rdn] [-x509] [-days n] [-set_serial n]
[-newhdr] [-addext ext] [-extensions section] [-reqexts section] [-precert] [-utf8]
[-nameopt] [-reqopt] [-subject] [-subj arg] [-sigopt nm:v] [-batch] [-verbose]
[-engine id]
```

DESCRIPTION

The req command primarily creates and processes certificate requests in PKCS#10 format. It can additionally create self signed certificates for use as root CAs for example.

OPTIONS

-help

Print out a usage message.

-inform DER|PEM

This specifies the input format. The DER option uses an ASN1 DER encoded form compatible with the PKCS#10. The PEM form is the default format: it consists of

the DER format base64 encoded with additional header and footer lines.

-outform DER|PEM

This specifies the output format, the options have the same meaning and default as the -inform option.

-in filename

This specifies the input filename to read a request from or standard input if this option is not specified. A request is only read if the creation options (-new and -newkey) are not specified.

-sigopt nm:v

Pass options to the signature algorithm during sign or verify operations. Names and values of these options are algorithm-specific.

-passin arg

The input file password source. For more information about the format of arg see the PASS PHRASE ARGUMENTS section in openssl(1).

-out filename

This specifies the output filename to write to or standard output by default.

-passout arg

The output file password source. For more information about the format of arg see the PASS PHRASE ARGUMENTS section in openssl(1).

-text

Prints out the certificate request in text form.

-subject

Prints out the request subject (or certificate subject if -x509 is specified)

-pubkey

Outputs the public key.

-noout

This option prevents output of the encoded version of the request.

-modulus

This option prints out the value of the modulus of the public key contained in the request.

-verify

Verifies the signature on the request.

-new

This option generates a new certificate request. It will prompt the user for the relevant field values. The actual fields prompted for and their maximum and minimum sizes are specified in the configuration file and any requested extensions.

If the `-key` option is not used it will generate a new RSA private key using information specified in the configuration file.

`-rand file...`

A file or files containing random data used to seed the random number generator. Multiple files can be specified separated by an OS-dependent character. The separator is `;` for MS-Windows, `,` for OpenVMS, and `:` for all others.

`[-writerand file]`

Writes random data to the specified file upon exit. This can be used with a subsequent `-rand` flag.

`-newkey arg`

This option creates a new certificate request and a new private key. The argument takes one of several forms. `rsa:nbits`, where `nbits` is the number of bits, generates an RSA key `nbits` in size. If `nbits` is omitted, i.e. `-newkey rsa` specified, the default key size, specified in the configuration file is used.

All other algorithms support the `-newkey alg:file` form, where `file` may be an algorithm parameter file, created by the `genpkey -genparam` command or and X.509 certificate for a key with appropriate algorithm.

`param:file` generates a key using the parameter file or certificate file, the algorithm is determined by the parameters. `alname:file` use algorithm `alname` and parameter file `file`: the two algorithms must match or an error occurs.

`alname` just uses algorithm `alname`, and parameters, if necessary should be specified via `-pkeyopt` parameter.

`dsa:filename` generates a DSA key using the parameters in the file `filename`.

`ec:filename` generates EC key (usable both with ECDSA or ECDH algorithms),

`gost2001:filename` generates GOST R 34.10-2001 key (requires `ccgost` engine configured in the configuration file). If just `gost2001` is specified a

parameter set should be specified by `-pkeyopt paramset:X`

`-pkeyopt opt:value`

Set the public key algorithm option `opt` to value. The precise set of options supported depends on the public key algorithm used and its implementation. See `KEY GENERATION OPTIONS` in the `genpkey` manual page for more details.

`-key filename`

This specifies the file to read the private key from. It also accepts PKCS#8 format private keys for PEM format files.

`-keyform PEM|DER`

The format of the private key file specified in the `-key` argument. PEM is the default.

`-keyout filename`

This gives the filename to write the newly created private key to. If this option is not specified then the filename present in the configuration file is used.

`-nodes`

If this option is specified then if a private key is created it will not be encrypted.

`-digest`

This specifies the message digest to sign the request. Any digest supported by the OpenSSL `dgst` command can be used. This overrides the digest algorithm specified in the configuration file.

Some public key algorithms may override this choice. For instance, DSA signatures always use SHA1, GOST R 34.10 signatures always use GOST R 34.11-94 (`-md_gost94`), Ed25519 and Ed448 never use any digest.

`-config filename`

This allows an alternative configuration file to be specified. Optional; for a description of the default value, see "COMMAND SUMMARY" in `openssl(1)`.

`-subj arg`

Sets subject name for new request or supersedes the subject name when processing a request. The `arg` must be formatted as `/type0=value0/type1=value1/type2=....`. Keyword characters may be escaped by `\` (backslash), and whitespace is retained. Empty values are permitted, but the corresponding type will not be included in the request.

`-multivalue-rdn`

This option causes the `-subj` argument to be interpreted with full support for multivalued RDNs. Example:

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

If `-multi-rdn` is not used then the UID value is `123456+CN=John Doe`.

`-x509`

This option outputs a self signed certificate instead of a certificate request.

This is typically used to generate a test certificate or a self signed root CA.

The extensions added to the certificate (if any) are specified in the configuration file. Unless specified using the `set_serial` option, a large random number will be used for the serial number.

If existing request is specified with the `-in` option, it is converted to the self signed certificate otherwise new request is created.

`-days n`

When the `-x509` option is being used this specifies the number of days to certify the certificate for, otherwise it is ignored. `n` should be a positive integer. The default is 30 days.

`-set_serial n`

Serial number to use when outputting a self signed certificate. This may be specified as a decimal value or a hex value if preceded by `0x`.

`-addext ext`

Add a specific extension to the certificate (if the `-x509` option is present) or certificate request. The argument must have the form of a key=value pair as it would appear in a config file.

This option can be given multiple times.

`-extensions section`

`-reqexts section`

These options specify alternative sections to include certificate extensions (if the `-x509` option is present) or certificate request extensions. This allows several different sections to be used in the same configuration file to specify requests for a variety of purposes.

`-precert`

A poison extension will be added to the certificate, making it a "pre-certificate" (see RFC6962). This can be submitted to Certificate Transparency

logs in order to obtain signed certificate timestamps (SCTs). These SCTs can then be embedded into the pre-certificate as an extension, before removing the poison and signing the certificate.

This implies the `-new` flag.

`-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.

`-nameopt` option

Option which determines how the subject or issuer names are displayed. The option argument can be a single option or multiple options separated by commas. Alternatively the `-nameopt` switch may be used more than once to set multiple options. See the `x509(1)` manual page for details.

`-reqopt`

Customise the output format used with `-text`. The option argument can be a single option or multiple options separated by commas.

See discussion of the `-certopt` parameter in the `x509(1)` command.

`-newhdr`

Adds the word `NEW` to the PEM file header and footer lines on the outputted request. Some software (Netscape certificate server) and some CAs need this.

`-batch`

Non-interactive mode.

`-verbose`

Print extra details about the operations being performed.

`-engine id`

Specifying an engine (by its unique id string) will cause `req` to attempt to obtain a functional reference to the specified engine, thus initialising it if needed. The engine will then be set as the default for all available algorithms.

`-keygen_engine id`

Specifies an engine (by its unique id string) which would be used for key generation operations.

CONFIGURATION FILE FORMAT

The configuration options are specified in the req section of the configuration file. As with all configuration files if no value is specified in the specific section (i.e. req) then the initial unnamed or default section is searched too.

The options available are described in detail below.

input_password output_password

The passwords for the input private key file (if present) and the output private key file (if one will be created). The command line options passin and passout override the configuration file values.

default_bits

Specifies the default key size in bits.

This option is used in conjunction with the -new option to generate a new key.

It can be overridden by specifying an explicit key size in the -newkey option.

The smallest accepted key size is 512 bits. If no key size is specified then 2048 bits is used.

default_keyfile

This is the default filename to write a private key to. If not specified the key is written to standard output. This can be overridden by the -keyout option.

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 white space then the short name followed by white space 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.

RANDFILE

At startup the specified file is loaded into the random number generator, and at exit 256 bytes will be written to it. It is used for private key generation.

encrypt_key

If this is set to no then if a private key is generated it is not encrypted.

This is equivalent to the -nodes command line option. For compatibility encrypt_rsa_key is an equivalent option.

default_md

This option specifies the digest algorithm to use. Any digest supported by the OpenSSL dgst command can be used. This option can be overridden on the command line. Certain signing algorithms (i.e. Ed25519 and Ed448) will ignore any digest that has been set.

string_mask

This option masks out the use of certain string types in certain fields. Most users will not need to change this option.

It can be set to several values default which is also the default option uses PrintableStrings, T61Strings and BMPStrings if the pkix value is used then only PrintableStrings and BMPStrings will be used. This follows the PKIX recommendation in RFC2459. If the utf8only option is used then only UTF8Strings will be used: this is the PKIX recommendation in RFC2459 after 2003. Finally the nombstr option just uses PrintableStrings and T61Strings: certain software has problems with BMPStrings and UTF8Strings: in particular Netscape.

req_extensions

This specifies the configuration file section containing a list of extensions to add to the certificate request. It can be overridden by the -reqexts command line switch. See the x509v3_config(5) manual page for details of the extension section format.

x509_extensions

This specifies the configuration file section containing a list of extensions to add to certificate generated when the -x509 switch is used. It can be overridden by the -extensions command line switch.

prompt

If set to the value no this disables prompting of certificate fields and just takes values from the config file directly. It also changes the expected format of the distinguished_name and attributes sections.

utf8

If set to the value yes then 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.

attributes

This specifies the section containing any request attributes: its format is the same as distinguished_name. Typically these may contain the challengePassword or unstructuredName types. They are currently ignored by OpenSSL's request signing utilities but some CAs might want them.

distinguished_name

This specifies the section containing the distinguished name fields to prompt for when generating a certificate or certificate request. The format is described in the next section.

DISTINGUISHED NAME AND ATTRIBUTE SECTION FORMAT

There are two separate formats for the distinguished name and attribute sections.

If the prompt option is set to no then these sections just consist of field names and values: for example,

```
CN=My Name
```

```
OU=My Organization
```

```
emailAddress=someone@somewhere.org
```

This allows external programs (e.g. GUI based) to generate a template file with all the field names and values and just pass it to req. An example of this kind of configuration file is contained in the EXAMPLES section.

Alternatively if the prompt option is absent or not set to no then the file contains field prompting information. It consists of lines of the form:

```
fieldName="prompt"
```

```
fieldName_default="default field value"
```

```
fieldName_min= 2
```

```
fieldName_max= 4
```

"fieldName" is the field name being used, for example commonName (or CN). The "prompt" string is used to ask the user to enter the relevant details. If the user enters nothing then the default value is used if no default value is present then the field is omitted. A field can still be omitted if a default value is present if

the user just enters the '.' character.

The number of characters entered must be between the `fieldName_min` and `fieldName_max` limits: there may be additional restrictions based on the field being used (for example `countryName` can only ever be two characters long and must fit in a `PrintableString`).

Some fields (such as `organizationName`) can be used more than once in a DN. This presents a problem because configuration files will not recognize the same name occurring twice. To avoid this problem if the `fieldName` contains some characters followed by a full stop they will be ignored. So for example a second `organizationName` can be input by calling it "1.organizationName".

The actual permitted field names are any object identifier short or long names.

These are compiled into OpenSSL and include the usual values such as `commonName`, `countryName`, `localityName`, `organizationName`, `organizationalUnitName`, `stateOrProvinceName`. Additionally `emailAddress` is included as well as `name`, `surname`, `givenName`, `initials`, and `dnQualifier`.

Additional object identifiers can be defined with the `oid_file` or `oid_section` options in the configuration file. Any additional fields will be treated as though they were a `DirectoryString`.

EXAMPLES

Examine and verify certificate request:

```
openssl req -in req.pem -text -verify -noout
```

Create a private key and then generate a certificate request from it:

```
openssl genrsa -out key.pem 2048
```

```
openssl req -new -key key.pem -out req.pem
```

The same but just using req:

```
openssl req -newkey rsa:2048 -keyout key.pem -out req.pem
```

Generate a self signed root certificate:

```
openssl req -x509 -newkey rsa:2048 -keyout key.pem -out req.pem
```

Example of a file pointed to by the `oid_file` option:

```
1.2.3.4    shortName    A longer Name
```

```
1.2.3.6    otherName    Other longer Name
```

Example of a section pointed to by `oid_section` making use of variable expansion:

```
testoid1=1.2.3.5
```

testoid2=\${testoid1}.6

Sample configuration file prompting for field values:

[req]

default_bits = 2048

default_keyfile = privkey.pem

distinguished_name = req_distinguished_name

attributes = req_attributes

req_extensions = v3_ca

dirstring_type = nobmp

[req_distinguished_name]

countryName = Country Name (2 letter code)

countryName_default = AU

countryName_min = 2

countryName_max = 2

localityName = Locality Name (eg, city)

organizationalUnitName = Organizational Unit Name (eg, section)

commonName = Common Name (eg, YOUR name)

commonName_max = 64

emailAddress = Email Address

emailAddress_max = 40

[req_attributes]

challengePassword = A challenge password

challengePassword_min = 4

challengePassword_max = 20

[v3_ca]

subjectKeyIdentifier=hash

authorityKeyIdentifier=keyid:always,issuer:always

basicConstraints = critical, CA:true

Sample configuration containing all field values:

RANDFILE = \$ENV::HOME/.rnd

[req]

default_bits = 2048

default_keyfile = keyfile.pem

distinguished_name = req_distinguished_name

attributes = req_attributes

prompt = no

output_password = mypass

[req_distinguished_name]

C = GB

ST = Test State or Province

L = Test Locality

O = Organization Name

OU = Organizational Unit Name

CN = Common Name

emailAddress = test@email.address

[req_attributes]

challengePassword = A challenge password

Example of giving the most common attributes (subject and extensions) on the command line:

```
openssl req -new -subj "/C=GB/CN=foo" \  
    -addext "subjectAltName = DNS:foo.co.uk" \  
    -addext "certificatePolicies = 1.2.3.4" \  
    -newkey rsa:2048 -keyout key.pem -out req.pem
```

NOTES

The header and footer lines in the PEM format are normally:

```
-----BEGIN CERTIFICATE REQUEST-----
```

```
-----END CERTIFICATE REQUEST-----
```

some software (some versions of Netscape certificate server) instead needs:

```
-----BEGIN NEW CERTIFICATE REQUEST-----
```

```
-----END NEW CERTIFICATE REQUEST-----
```

which is produced with the `-newhdr` option but is otherwise compatible. Either form is accepted transparently on input.

The certificate requests generated by Xenroll with MSIE have extensions added. It includes the `keyUsage` extension which determines the type of key (signature only or general purpose) and any additional OIDs entered by the script in an `extendedKeyUsage` extension.

DIAGNOSTICS

The following messages are frequently asked about:

Using configuration from /some/path/openssl.cnf

Unable to load config info

This is followed some time later by...

unable to find 'distinguished_name' in config

problems making Certificate Request

The first error message is the clue: it can't find the configuration file! Certain operations (like examining a certificate request) don't need a configuration file so its use isn't enforced. Generation of certificates or requests however does need a configuration file. This could be regarded as a bug.

Another puzzling message is this:

Attributes:

a0:00

this is displayed when no attributes are present and the request includes the correct empty SET OF structure (the DER encoding of which is 0xa0 0x00). If you just see:

Attributes:

then the SET OF is missing and the encoding is technically invalid (but it is tolerated). See the description of the command line option `-asn1-kludge` for more information.

BUGS

OpenSSL's handling of T61Strings (aka TeletexStrings) is broken: it effectively treats them as ISO-8859-1 (Latin 1), Netscape and MSIE have similar behaviour.

This can cause problems if you need characters that aren't available in PrintableStrings and you don't want to or can't use BMPStrings.

As a consequence of the T61String handling the only correct way to represent accented characters in OpenSSL is to use a BMPString: unfortunately Netscape currently chokes on these. If you have to use accented characters with Netscape and MSIE then you currently need to use the invalid T61String form.

The current prompting is not very friendly. It doesn't allow you to confirm what you've just entered. Other things like extensions in certificate requests are statically defined in the configuration file. Some of these: like an email address

in subjectAltName should be input by the user.

SEE ALSO

x509(1), ca(1), genrsa(1), gendsa(1), config(5), x509v3_config(5)

COPYRIGHT

Copyright 2000-2019 The OpenSSL Project Authors. All Rights Reserved.

Licensed under the OpenSSL license (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>>.

1.1.1f

2023-02-06

REQ(1SSL)