## NAME

openssl−x509, x509 − Certificate display and signing utility

## SYNOPSIS

**openssl x509** [**−help**] [**−inform DER|PEM**] [**−outform DER|PEM**] [**−keyform DER|PEM**] [**−CAform DER|PEM**] [**−CAkeyform DER|PEM**] [**−in filename**] [**−out filename**] [**−serial**] [**−hash**] [**−subject_hash**] [**−issuer_hash**] [**−ocspid**] [**−subject**] [**−issuer**] [**−nameopt option**] [**−email**] [**−ocsp_uri**] [**−startdate**] [**−enddate**] [**−purpose**] [**−dates**] [**−checkend num**] [**−modulus**] [**−pubkey**] [**−fingerprint**] [**−alias**] [**−noout**] [**−trustout**] [**−clrtrust**] [**−clrreject**] [**−addtrust arg**] [**−addreject arg**] [**−setalias arg**] [**−days arg**] [**−set_serial n**] [**−signkey filename**] [**−passin arg**] [**−x509toreq**] [**−req**] [**−CA filename**] [**−CAkey filename**] [**−CAcreateserial**] [**−CAserial filename**] [**−force_pubkey key**] [**−text**] [**−ext extensions**] [**−certopt option**] [**−C**] [***−digest***] [**−clrext**] [**−extfile filename**] [**−extensions section**] [**−rand file...**] [**−writerand file**] [**−engine id**] [**−preserve_dates**]

## DESCRIPTION

The **x509** command is a multi purpose certificate utility. It can be used to display certificate information, convert certificates to various forms, sign certificate requests like a ‘‘mini CA’’ or edit certificate trust settings.

Since there are a large number of options they will split up into various sections.

## OPTIONS

### Input, Output, and General Purpose Options

**−help**

Print out a usage message.

**−inform DER|PEM**

This specifies the input format normally the command will expect an X509 certificate but this can change if other options such as **−req** are present. The DER format is the DER encoding of the certificate and PEM is the base64 encoding of the DER encoding with header and footer lines added. The default format is PEM.

**−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 certificate from or standard input if this option is not specified.

**−out filename**

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

***−digest***

The digest to use. This affects any signing or display option that uses a message digest, such as the **−fingerprint**, **−signkey** and **−CA** options. Any digest supported by the OpenSSL **dgst** command can be used. If not specified then SHA1 is used with **−fingerprint** or the default digest for the signing algorithm is used, typically SHA256.

**−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.

**−engine id**

Specifying an engine (by its unique **id** string) will cause **x509** 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.

**−preserve_dates**
When signing a certificate, preserve the ‘‘notBefore’’ and ‘‘notAfter’’ dates instead of adjusting them to current time and duration. Cannot be used with the **−days** option.

**Display Options**
Note: the **−alias** and **−purpose** options are also display options but are described in the **TRUST SETTINGS** section.

**−text**
Prints out the certificate in text form. Full details are output including the public key, signature algorithms, issuer and subject names, serial number any extensions present and any trust settings.

**−ext extensions**
Prints out the certificate extensions in text form. Extensions are specified with a comma separated string, e.g., ‘‘subjectAltName,subjectKeyIdentifier’’. See the **x509v3_config**(5) manual page for the extension names.

**−certopt option**
Customise the output format used with **−text**. The **option** argument can be a single option or multiple options separated by commas. The **−certopt** switch may be also be used more than once to set multiple options. See the **TEXT OPTIONS** section for more information.

**−noout**
This option prevents output of the encoded version of the certificate.

**−pubkey**
Outputs the certificate’s SubjectPublicKeyInfo block in PEM format.

**−modulus**
This option prints out the value of the modulus of the public key contained in the certificate.

**−serial**
Outputs the certificate serial number.

**−subject_hash**
Outputs the ‘‘hash’’ of the certificate subject name. This is used in OpenSSL to form an index to allow certificates in a directory to be looked up by subject name.

**−issuer_hash**
Outputs the ‘‘hash’’ of the certificate issuer name.

**−ocspid**
Outputs the OCSP hash values for the subject name and public key.

**−hash**
Synonym for ‘‘−subject_hash’’ for backward compatibility reasons.

**−subject_hash_old**
Outputs the ‘‘hash’’ of the certificate subject name using the older algorithm as used by OpenSSL before version 1.0.0.

**−issuer_hash_old**
Outputs the ‘‘hash’’ of the certificate issuer name using the older algorithm as used by OpenSSL before version 1.0.0.

**−subject**
Outputs the subject name.

**−issuer**
Outputs the issuer name.

**−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 **NAME OPTIONS** section for more information.

**−email**
Outputs the email address(es) if any.

**−ocsp_uri**
Outputs the OCSP responder address(es) if any.

**−startdate**
Prints out the start date of the certificate, that is the notBefore date.

**−enddate**
Prints out the expiry date of the certificate, that is the notAfter date.

**−dates**
Prints out the start and expiry dates of a certificate.

**−checkend arg**
Checks if the certificate expires within the next **arg** seconds and exits non-zero if yes it will expire or zero if not.

**−fingerprint**
Calculates and outputs the digest of the DER encoded version of the entire certificate (see digest options). This is commonly called a ‘‘fingerprint’’. Because of the nature of message digests, the fingerprint of a certificate is unique to that certificate and two certificates with the same fingerprint can be considered to be the same.

**−C** This outputs the certificate in the form of a C source file.

**Trust Settings**

A **trusted certificate** is an ordinary certificate which has several additional pieces of information attached to it such as the permitted and prohibited uses of the certificate and an ‘‘alias’’.

Normally when a certificate is being verified at least one certificate must be ‘‘trusted’’. By default a trusted certificate must be stored locally and must be a root CA: any certificate chain ending in this CA is then usable for any purpose.

Trust settings currently are only used with a root CA. They allow a finer control over the purposes the root CA can be used for. For example a CA may be trusted for SSL client but not SSL server use.

See the description of the **verify** utility for more information on the meaning of trust settings.

Future versions of OpenSSL will recognize trust settings on any certificate: not just root CAs.

**−trustout**
This causes **x509** to output a **trusted** certificate. An ordinary or trusted certificate can be input but by default an ordinary certificate is output and any trust settings are discarded. With the **−trustout** option a trusted certificate is output. A trusted certificate is automatically output if any trust settings are modified.

**−setalias arg**
Sets the alias of the certificate. This will allow the certificate to be referred to using a nickname for example ‘‘Steve’s Certificate’’.

**−alias**
Outputs the certificate alias, if any.

**−clrtrust**
Clears all the permitted or trusted uses of the certificate.

**−clrreject**
Clears all the prohibited or rejected uses of the certificate.

**−addtrust arg**
Adds a trusted certificate use. Any object name can be used here but currently only **clientAuth** (SSL client use), **serverAuth** (SSL server use), **emailProtection** (S/MIME email) and **anyExtendedKeyUsage** are used. As of OpenSSL 1.1.0, the last of these blocks all purposes when

rejected or enables all purposes when trusted. Other OpenSSL applications may define additional uses.

**−addreject arg**
Adds a prohibited use. It accepts the same values as the **−addtrust** option.

**−purpose**
This option performs tests on the certificate extensions and outputs the results. For a more complete description see the **CERTIFICATE EXTENSIONS** section.

**Signing Options**

The **x509** utility can be used to sign certificates and requests: it can thus behave like a "mini CA".

**−signkey filename**
This option causes the input file to be self signed using the supplied private key.

If the input file is a certificate it sets the issuer name to the subject name (i.e. makes it self signed) changes the public key to the supplied value and changes the start and end dates. The start date is set to the current time and the end date is set to a value determined by the **−days** option. Any certificate extensions are retained unless the **−clrext** option is supplied; this includes, for example, any existing key identifier extensions.

If the input is a certificate request then a self signed certificate is created using the supplied private key using the subject name in the request.

**−passin arg**
The key password source. For more information about the format of **arg** see the **PASS PHRASE ARGUMENTS** section in **openssl**(1).

**−clrext**
Delete any extensions from a certificate. This option is used when a certificate is being created from another certificate (for example with the **−signkey** or the **−CA** options). Normally all extensions are retained.

**−keyform PEM|DER**
Specifies the format (DER or PEM) of the private key file used in the **−signkey** option.

**−days arg**
Specifies the number of days to make a certificate valid for. The default is 30 days. Cannot be used with the **−preserve_dates** option.

**−x509toreq**
Converts a certificate into a certificate request. The **−signkey** option is used to pass the required private key.

**−req**
By default a certificate is expected on input. With this option a certificate request is expected instead.

**−set_serial n**
Specifies the serial number to use. This option can be used with either the **−signkey** or **−CA** options. If used in conjunction with the **−CA** option the serial number file (as specified by the **−CAserial** or **−CAcreateserial** options) is not used.

The serial number can be decimal or hex (if preceded by **0x**).

**−CA filename**
Specifies the CA certificate to be used for signing. When this option is present **x509** behaves like a "mini CA". The input file is signed by this CA using this option: that is its issuer name is set to the subject name of the CA and it is digitally signed using the CAs private key.

This option is normally combined with the **−req** option. Without the **−req** option the input is a certificate which must be self signed.

**−CAkey filename**
Sets the CA private key to sign a certificate with. If this option is not specified then it is assumed that the CA private key is present in the CA certificate file.

**−CAserial filename**
Sets the CA serial number file to use.

When the **−CA** option is used to sign a certificate it uses a serial number specified in a file. This file consists of one line containing an even number of hex digits with the serial number to use. After each use the serial number is incremented and written out to the file again.

The default filename consists of the CA certificate file base name with ".srl" appended. For example if the CA certificate file is called "mycacert.pem" it expects to find a serial number file called "mycacert.srl".

**−CAcreateserial**
With this option the CA serial number file is created if it does not exist: it will contain the serial number "02" and the certificate being signed will have the 1 as its serial number. If the **−CA** option is specified and the serial number file does not exist a random number is generated; this is the recommended practice.

**−extfile filename**
File containing certificate extensions to use. If not specified then no extensions are added to the certificate.

**−extensions section**
The section to add certificate extensions from. If this option is not specified then the extensions should either be contained in the unnamed (default) section or the default section should contain a variable called "extensions" which contains the section to use. See the **x509v3_config**(5) manual page for details of the extension section format.

**−force_pubkey key**
When a certificate is created set its public key to **key** instead of the key in the certificate or certificate request. This option is useful for creating certificates where the algorithm can't normally sign requests, for example DH.

The format or **key** can be specified using the **−keyform** option.

**Name Options**
The **nameopt** command line switch determines how the subject and issuer names are displayed. If no **nameopt** switch is present the default "oneline" format is used which is compatible with previous versions of OpenSSL. Each option is described in detail below, all options can be preceded by a **−** to turn the option off. Only the first four will normally be used.

**compat**
Use the old format.

**RFC2253**
Displays names compatible with RFC2253 equivalent to **esc_2253**, **esc_ctrl**, **esc_msb**, **utf8**, **dump_nostr**, **dump_unknown**, **dump_der**, **sep_comma_plus**, **dn_rev** and **sname**.

**oneline**
A oneline format which is more readable than RFC2253. It is equivalent to specifying the **esc_2253**, **esc_ctrl**, **esc_msb**, **utf8**, **dump_nostr**, **dump_der**, **use_quote**, **sep_comma_plus_space**, **space_eq** and **sname** options. This is the *default* of no name options are given explicitly.

**multiline**
A multiline format. It is equivalent **esc_ctrl**, **esc_msb**, **sep_multiline**, **space_eq**, **lname** and **align**.

**esc_2253**
Escape the "special" characters required by RFC2253 in a field. That is **,+"<>;**. Additionally **#** is escaped at the beginning of a string and a space character at the beginning or end of a string.

**esc_2254**
Escape the ‘‘special’’ characters required by RFC2254 in a field. That is the **NUL** character as well as and ()*.

**esc_ctrl**
Escape control characters. That is those with ASCII values less than 0x20 (space) and the delete (0x7f) character. They are escaped using the RFC2253 \XX notation (where XX are two hex digits representing the character value).

**esc_msb**
Escape characters with the MSB set, that is with ASCII values larger than 127.

**use_quote**
Escapes some characters by surrounding the whole string with **"** characters, without the option all escaping is done with the **** character.

**utf8**
Convert all strings to UTF8 format first. This is required by RFC2253. If you are lucky enough to have a UTF8 compatible terminal then the use of this option (and **not** setting **esc_msb**) may result in the correct display of multibyte (international) characters. Is this option is not present then multibyte characters larger than 0xff will be represented using the format \UXXXX for 16 bits and \WXXXXXXXX for 32 bits. Also if this option is off any UTF8Strings will be converted to their character form first.

**ignore_type**
This option does not attempt to interpret multibyte characters in any way. That is their content octets are merely dumped as though one octet represents each character. This is useful for diagnostic purposes but will result in rather odd looking output.

**show_type**
Show the type of the ASN1 character string. The type precedes the field contents. For example ‘‘BMPSTRING: Hello World’’.

**dump_der**
When this option is set any fields that need to be hexdumped will be dumped using the DER encoding of the field. Otherwise just the content octets will be displayed. Both options use the RFC2253 **#XXXX...** format.

**dump_nostr**
Dump non character string types (for example OCTET STRING) if this option is not set then non character string types will be displayed as though each content octet represents a single character.

**dump_all**
Dump all fields. This option when used with **dump_der** allows the DER encoding of the structure to be unambiguously determined.

**dump_unknown**
Dump any field whose OID is not recognised by OpenSSL.

**sep_comma_plus**, **sep_comma_plus_space**, **sep_semi_plus_space**, **sep_multiline**
These options determine the field separators. The first character is between RDNs and the second between multiple AVAs (multiple AVAs are very rare and their use is discouraged). The options ending in ‘‘space’’ additionally place a space after the separator to make it more readable. The **sep_multiline** uses a linefeed character for the RDN separator and a spaced **+** for the AVA separator. It also indents the fields by four characters. If no field separator is specified then **sep_comma_plus_space** is used by default.

**dn_rev**
Reverse the fields of the DN. This is required by RFC2253. As a side effect this also reverses the order of multiple AVAs but this is permissible.

**nofname**, **sname**, **lname**, **oid**
These options alter how the field name is displayed. **nofname** does not display the field at all. **sname** uses the ‘‘short name’’ form (CN for commonName for example). **lname** uses the long form. **oid** represents the OID in numerical form and is useful for diagnostic purpose.

**align**
Align field values for a more readable output. Only usable with **sep_multiline**.

**space_eq**
Places spaces round the **=** character which follows the field name.

**Text Options**
As well as customising the name output format, it is also possible to customise the actual fields printed using the **certopt** options when the **text** option is present. The default behaviour is to print all fields.

**compatible**
Use the old format. This is equivalent to specifying no output options at all.

**no_header**
Don’t print header information: that is the lines saying ‘‘Certificate’’ and ‘‘Data’’.

**no_version**
Don’t print out the version number.

**no_serial**
Don’t print out the serial number.

**no_signame**
Don’t print out the signature algorithm used.

**no_validity**
Don’t print the validity, that is the **notBefore** and **notAfter** fields.

**no_subject**
Don’t print out the subject name.

**no_issuer**
Don’t print out the issuer name.

**no_pubkey**
Don’t print out the public key.

**no_sigdump**
Don’t give a hexadecimal dump of the certificate signature.

**no_aux**
Don’t print out certificate trust information.

**no_extensions**
Don’t print out any X509V3 extensions.

**ext_default**
Retain default extension behaviour: attempt to print out unsupported certificate extensions.

**ext_error**
Print an error message for unsupported certificate extensions.

**ext_parse**
ASN1 parse unsupported extensions.

**ext_dump**
Hex dump unsupported extensions.

**ca_default**
The value used by the **ca** utility, equivalent to **no_issuer**, **no_pubkey**, **no_header**, and **no_version**.

## EXAMPLES

Note: in these examples the '\' means the example should be all on one line.

Display the contents of a certificate:

```
openssl x509 -in cert.pem -noout -text
```

Display the "Subject Alternative Name" extension of a certificate:

```
openssl x509 -in cert.pem -noout -ext subjectAltName
```

Display more extensions of a certificate:

```
openssl x509 -in cert.pem -noout -ext subjectAltName,nsCertType
```

Display the certificate serial number:

```
openssl x509 -in cert.pem -noout -serial
```

Display the certificate subject name:

```
openssl x509 -in cert.pem -noout -subject
```

Display the certificate subject name in RFC2253 form:

```
openssl x509 -in cert.pem -noout -subject -nameopt RFC2253
```

Display the certificate subject name in oneline form on a terminal supporting UTF8:

```
openssl x509 -in cert.pem -noout -subject -nameopt oneline,-esc_msb
```

Display the certificate SHA1 fingerprint:

```
openssl x509 -sha1 -in cert.pem -noout -fingerprint
```

Convert a certificate from PEM to DER format:

```
openssl x509 -in cert.pem -inform PEM -out cert.der -outform DER
```

Convert a certificate to a certificate request:

```
openssl x509 -x509toreq -in cert.pem -out req.pem -signkey key.pem
```

Convert a certificate request into a self signed certificate using extensions for a CA:

```
openssl x509 -req -in careq.pem -extfile openssl.cnf -extensions v3_ca \
       -signkey key.pem -out cacert.pem
```

Sign a certificate request using the CA certificate above and add user certificate extensions:

```
openssl x509 -req -in req.pem -extfile openssl.cnf -extensions v3_usr \
       -CA cacert.pem -CAkey key.pem -CAcreateserial
```

Set a certificate to be trusted for SSL client use and change set its alias to "Steve's Class 1 CA"

```
openssl x509 -in cert.pem -addtrust clientAuth \
       -setalias "Steve's Class 1 CA" -out trust.pem
```

## NOTES

The PEM format uses the header and footer lines:

```
-----BEGIN CERTIFICATE-----
-----END CERTIFICATE-----
```

it will also handle files containing:

```
-----BEGIN X509 CERTIFICATE-----
-----END X509 CERTIFICATE-----
```

Trusted certificates have the lines

```
-----BEGIN TRUSTED CERTIFICATE-----
-----END TRUSTED CERTIFICATE-----
```

The conversion to UTF8 format used with the name options assumes that T61Strings use the ISO8859–1 character set. This is wrong but Netscape and MSIE do this as do many certificates. So although this is incorrect it is more likely to display the majority of certificates correctly.

The **–email** option searches the subject name and the subject alternative name extension. Only unique email addresses will be printed out: it will not print the same address more than once.

## CERTIFICATE EXTENSIONS

The **–purpose** option checks the certificate extensions and determines what the certificate can be used for. The actual checks done are rather complex and include various hacks and workarounds to handle broken certificates and software.

The same code is used when verifying untrusted certificates in chains so this section is useful if a chain is rejected by the verify code.

The basicConstraints extension CA flag is used to determine whether the certificate can be used as a CA. If the CA flag is true then it is a CA, if the CA flag is false then it is not a CA. **All** CAs should have the CA flag set to true.

If the basicConstraints extension is absent then the certificate is considered to be a "possible CA" other extensions are checked according to the intended use of the certificate. A warning is given in this case because the certificate should really not be regarded as a CA: however it is allowed to be a CA to work around some broken software.

If the certificate is a V1 certificate (and thus has no extensions) and it is self signed it is also assumed to be a CA but a warning is again given: this is to work around the problem of Verisign roots which are V1 self signed certificates.

If the keyUsage extension is present then additional restraints are made on the uses of the certificate. A CA certificate **must** have the keyCertSign bit set if the keyUsage extension is present.

The extended key usage extension places additional restrictions on the certificate uses. If this extension is present (whether critical or not) the key can only be used for the purposes specified.

A complete description of each test is given below. The comments about basicConstraints and keyUsage and V1 certificates above apply to **all** CA certificates.

**SSL Client**
: The extended key usage extension must be absent or include the "web client authentication" OID. keyUsage must be absent or it must have the digitalSignature bit set. Netscape certificate type must be absent or it must have the SSL client bit set.

**SSL Client CA**
: The extended key usage extension must be absent or include the "web client authentication" OID. Netscape certificate type must be absent or it must have the SSL CA bit set: this is used as a work around if the basicConstraints extension is absent.

**SSL Server**
: The extended key usage extension must be absent or include the "web server authentication" and/or one of the SGC OIDs. keyUsage must be absent or it must have the digitalSignature, the keyEncipherment set or both bits set. Netscape certificate type must be absent or have the SSL server bit set.

**SSL Server CA**
: The extended key usage extension must be absent or include the "web server authentication" and/or one of the SGC OIDs. Netscape certificate type must be absent or the SSL CA bit must be set: this is used as a work around if the basicConstraints extension is absent.

**Netscape SSL Server**
: For Netscape SSL clients to connect to an SSL server it must have the keyEncipherment bit set if the keyUsage extension is present. This isn't always valid because some cipher suites use the key for digital signing. Otherwise it is the same as a normal SSL server.

**Common S/MIME Client Tests**
The extended key usage extension must be absent or include the "email protection" OID. Netscape certificate type must be absent or should have the S/MIME bit set. If the S/MIME bit is not set in Netscape certificate type then the SSL client bit is tolerated as an alternative but a warning is shown: this is because some Verisign certificates don't set the S/MIME bit.

**S/MIME Signing**
In addition to the common S/MIME client tests the digitalSignature bit or the nonRepudiation bit must be set if the keyUsage extension is present.

**S/MIME Encryption**
In addition to the common S/MIME tests the keyEncipherment bit must be set if the keyUsage extension is present.

**S/MIME CA**
The extended key usage extension must be absent or include the "email protection" OID. Netscape certificate type must be absent or must have the S/MIME CA bit set: this is used as a work around if the basicConstraints extension is absent.

**CRL Signing**
The keyUsage extension must be absent or it must have the CRL signing bit set.

**CRL Signing CA**
The normal CA tests apply. Except in this case the basicConstraints extension must be present.

## BUGS

Extensions in certificates are not transferred to certificate requests and vice versa.

It is possible to produce invalid certificates or requests by specifying the wrong private key or using inconsistent options in some cases: these should be checked.

There should be options to explicitly set such things as start and end dates rather than an offset from the current time.

## SEE ALSO

**req** (1), **ca** (1), **genrsa** (1), **gendsa** (1), **verify** (1), **x509v3_config** (5)

## HISTORY

The hash algorithm used in the **−subject_hash** and **−issuer_hash** options before OpenSSL 1.0.0 was based on the deprecated MD5 algorithm and the encoding of the distinguished name. In OpenSSL 1.0.0 and later it is based on a canonical version of the DN using SHA1. This means that any directories using the old form must have their links rebuilt using **c_rehash** or similar.

## COPYRIGHT