NAME

openssl - OpenSSL command line program

SYNOPSIS

openssl *command* [*options* ...] [*parameters* ...]

openssl no-*XXX* [*options*]

DESCRIPTION

OpenSSL is a cryptography toolkit implementing the Secure Sockets Layer (SSL v2/v3) and Transport Layer Security (TLS v1) network protocols and related cryptography standards required by them.

The **openssl** program is a command line program for using the various cryptography functions of OpenSSL's **crypto** library from the shell. It can be used for

- Creation and management of private keys, public keys and parameters
- Public key cryptographic operations
- Creation of X.509 certificates, CSRs and CRLs
- Calculation of Message Digests and Message Authentication Codes
- Encryption and Decryption with Ciphers
- SSL/TLS Client and Server Tests
- Handling of S/MIME signed or encrypted mail
- Timestamp requests, generation and verification

COMMAND SUMMARY

The **openssl** program provides a rich variety of commands (*command* in the "SYNOPSIS" above). Each command can have many options and argument parameters, shown above as *options* and *parameters*.

Detailed documentation and use cases for most standard subcommands are available (e.g., **openssl-x509**(1)). The subcommand **openssl-list**(1) may be used to list subcommands.

The command **no-***XXX* tests whether a command of the specified name is available. If no command named *XXX* exists, it returns 0 (success) and prints **no-***XXX*; otherwise it returns 1 and prints *XXX*. In both cases, the output goes to **stdout** and nothing is printed to **stderr**. Additional command line arguments are always ignored. Since for each cipher there is a command of the same name, this provides an easy way for shell scripts to test for the availability of ciphers in the **openssl** program. (**no-***XXX* is not able to detect pseudo-commands such as **quit**, **list**, or **no-***XXX* itself.)

Configuration Option

Many commands use an external configuration file for some or all of their arguments and have a **-config** option to specify that file. The default name of the file is *openssl.cnf* in the default certificate storage area, which can be determined from the **openssl-version**(1) command using the **-d** or **-a** option. The environment variable **OPENSSL_CONF** can be used to specify a different file location or to disable loading a configuration (using the empty string).

Among others, the configuration file can be used to load modules and to specify parameters for generating certificates and random numbers. See **config**(5) for details.

Standard Commands

asn1parse
: Parse an ASN.1 sequence.

ca Certificate Authority (CA) Management.

ciphers
: Cipher Suite Description Determination.

cms CMS (Cryptographic Message Syntax) command.

crl Certificate Revocation List (CRL) Management.

crl2pkcs7
: CRL to PKCS#7 Conversion.

dgst Message Digest calculation. MAC calculations are superseded by **openssl-mac**(1).

dhparam
: Generation and Management of Diffie-Hellman Parameters. Superseded by **openssl-genpkey**(1) and **openssl-pkeyparam**(1).

dsa DSA Data Management.

dsaparam
: DSA Parameter Generation and Management. Superseded by **openssl-genpkey**(1) and **openssl-pkeyparam**(1).

ec EC (Elliptic curve) key processing.

ecparam

EC parameter manipulation and generation.

enc Encryption, decryption, and encoding.

engine
Engine (loadable module) information and manipulation.

errstr
Error Number to Error String Conversion.

fipsinstall
FIPS configuration installation.

gendsa
Generation of DSA Private Key from Parameters. Superseded by **openssl-genpkey**(1) and **openssl-pkey**(1).

genpkey
Generation of Private Key or Parameters.

genrsa
Generation of RSA Private Key. Superseded by **openssl-genpkey**(1).

help Display information about a command’s options.

info Display diverse information built into the OpenSSL libraries.

kdf Key Derivation Functions.

list List algorithms and features.

mac Message Authentication Code Calculation.

nseq
Create or examine a Netscape certificate sequence.

ocsp
Online Certificate Status Protocol command.

passwd

Generation of hashed passwords.

pkcs12
PKCS#12 Data Management.

pkcs7
PKCS#7 Data Management.

pkcs8
PKCS#8 format private key conversion command.

pkey
Public and private key management.

pkeyparam
Public key algorithm parameter management.

pkeyutl
Public key algorithm cryptographic operation command.

prime
Compute prime numbers.

rand
Generate pseudo-random bytes.

rehash
Create symbolic links to certificate and CRL files named by the hash values.

req PKCS#10 X.509 Certificate Signing Request (CSR) Management.

rsa RSA key management.

rsautl
RSA command for signing, verification, encryption, and decryption. Superseded by **openssl-pkeyutl**(1).

s_client
This implements a generic SSL/TLS client which can establish a transparent connection to a remote server speaking SSL/TLS. It’s intended for testing purposes only and provides only

rudimentary interface functionality but internally uses mostly all functionality of the OpenSSL **ssl** library.

s_server
: This implements a generic SSL/TLS server which accepts connections from remote clients speaking SSL/TLS. It's intended for testing purposes only and provides only rudimentary interface functionality but internally uses mostly all functionality of the OpenSSL **ssl** library. It provides both an own command line oriented protocol for testing SSL functions and a simple HTTP response facility to emulate an SSL/TLS-aware webserver.

s_time
: SSL Connection Timer.

sess_id
: SSL Session Data Management.

smime
: S/MIME mail processing.

speed
: Algorithm Speed Measurement.

spkac
: SPKAC printing and generating command.

srp Maintain SRP password file. This command is deprecated.

storeutl
: Command to list and display certificates, keys, CRLs, etc.

ts Time Stamping Authority command.

verify
: X.509 Certificate Verification. See also the **openssl-verification-options**(1) manual page.

version
: OpenSSL Version Information.

x509
: X.509 Certificate Data Management.

Message Digest Commands

blake2b512

BLAKE2b-512 Digest

blake2s256

BLAKE2s-256 Digest

md2

MD2 Digest

md4

MD4 Digest

md5

MD5 Digest

mdc2

MDC2 Digest

rmd160

RMD-160 Digest

sha1

SHA-1 Digest

sha224

SHA-2 224 Digest

sha256

SHA-2 256 Digest

sha384

SHA-2 384 Digest

sha512

SHA-2 512 Digest

sha3-224

SHA-3 224 Digest

sha3-256
SHA-3 256 Digest

sha3-384
SHA-3 384 Digest

sha3-512
SHA-3 512 Digest

shake128
SHA-3 SHAKE128 Digest

shake256
SHA-3 SHAKE256 Digest

sm3 SM3 Digest

Encryption, Decryption, and Encoding Commands

The following aliases provide convenient access to the most used encodings and ciphers.

Depending on how OpenSSL was configured and built, not all ciphers listed here may be present. See **openssl-enc**(1) for more information.

aes128, **aes-128-cbc**, **aes-128-cfb**, **aes-128-ctr**, **aes-128-ecb**, **aes-128-ofb**
AES-128 Cipher

aes192, **aes-192-cbc**, **aes-192-cfb**, **aes-192-ctr**, **aes-192-ecb**, **aes-192-ofb**
AES-192 Cipher

aes256, **aes-256-cbc**, **aes-256-cfb**, **aes-256-ctr**, **aes-256-ecb**, **aes-256-ofb**
AES-256 Cipher

aria128, **aria-128-cbc**, **aria-128-cfb**, **aria-128-ctr**, **aria-128-ecb**, **aria-128-ofb**
Aria-128 Cipher

aria192, **aria-192-cbc**, **aria-192-cfb**, **aria-192-ctr**, **aria-192-ecb**, **aria-192-ofb**
Aria-192 Cipher

aria256, **aria-256-cbc**, **aria-256-cfb**, **aria-256-ctr**, **aria-256-ecb**, **aria-256-ofb**
Aria-256 Cipher

base64
Base64 Encoding

bf, **bf-cbc**, **bf-cfb**, **bf-ecb**, **bf-ofb**
Blowfish Cipher

camellia128, **camellia-128-cbc**, **camellia-128-cfb**, **camellia-128-ctr**, **camellia-128-ecb**, **camellia-128-ofb**
Camellia-128 Cipher

camellia192, **camellia-192-cbc**, **camellia-192-cfb**, **camellia-192-ctr**, **camellia-192-ecb**, **camellia-192-ofb**
Camellia-192 Cipher

camellia256, **camellia-256-cbc**, **camellia-256-cfb**, **camellia-256-ctr**, **camellia-256-ecb**, **camellia-256-ofb**
Camellia-256 Cipher

cast, **cast-cbc**
CAST Cipher

cast5-cbc, **cast5-cfb**, **cast5-ecb**, **cast5-ofb**
CAST5 Cipher

chacha20
Chacha20 Cipher

des, **des-cbc**, **des-cfb**, **des-ecb**, **des-ede**, **des-ede-cbc**, **des-ede-cfb**, **des-ede-ofb**, **des-ofb**
DES Cipher

des3, **desx**, **des-ede3**, **des-ede3-cbc**, **des-ede3-cfb**, **des-ede3-ofb**
Triple-DES Cipher

idea, **idea-cbc**, **idea-cfb**, **idea-ecb**, **idea-ofb**
IDEA Cipher

rc2, **rc2-cbc**, **rc2-cfb**, **rc2-ecb**, **rc2-ofb**
RC2 Cipher

rc4 RC4 Cipher

rc5, **rc5-cbc**, **rc5-cfb**, **rc5-ecb**, **rc5-ofb**
RC5 Cipher

seed, **seed-cbc**, **seed-cfb**, **seed-ecb**, **seed-ofb**
SEED Cipher

sm4, **sm4-cbc**, **sm4-cfb**, **sm4-ctr**, **sm4-ecb**, **sm4-ofb**
SM4 Cipher

OPTIONS

Details of which options are available depend on the specific command. This section describes some common options with common behavior.

Common Options

-help

Provides a terse summary of all options. If an option takes an argument, the "type" of argument is also given.

-- This terminates the list of options. It is mostly useful if any filename parameters start with a minus sign:

```
openssl verify [flags...] -- -cert1.pem...
```

Format Options

See **openssl-format-options**(1) for manual page.

Pass Phrase Options

See the **openssl-passphrase-options**(1) manual page.

Random State Options

Prior to OpenSSL 1.1.1, it was common for applications to store information about the state of the random-number generator in a file that was loaded at startup and rewritten upon exit. On modern operating systems, this is generally no longer necessary as OpenSSL will seed itself from a trusted entropy source provided by the operating system. These flags are still supported for special platforms or circumstances that might require them.

It is generally an error to use the same seed file more than once and every use of **-rand** should be paired with **-writerand**.

-rand *files*

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. Another way to specify multiple files is to repeat this flag with different filenames.

-writerand *file*

Writes the seed data to the specified *file* upon exit. This file can be used in a subsequent command invocation.

Certificate Verification Options

See the **openssl-verification-options**(1) manual page.

Name Format Options

See the **openssl-namedisplay-options**(1) manual page.

TLS Version Options

Several commands use SSL, TLS, or DTLS. By default, the commands use TLS and clients will offer the lowest and highest protocol version they support, and servers will pick the highest version that the client offers that is also supported by the server.

The options below can be used to limit which protocol versions are used, and whether TCP (SSL and TLS) or UDP (DTLS) is used. Note that not all protocols and flags may be available, depending on how OpenSSL was built.

-ssl3, **-tls1**, **-tls1_1**, **-tls1_2**, **-tls1_3**, **-no_ssl3**, **-no_tls1**, **-no_tls1_1**, **-no_tls1_2**, **-no_tls1_3**

These options require or disable the use of the specified SSL or TLS protocols. When a specific TLS version is required, only that version will be offered or accepted. Only one specific protocol can be given and it cannot be combined with any of the **no_** options. The **no_*** options do not work with **s_time** and **ciphers** commands but work with **s_client** and **s_server** commands.

-dtls, **-dtls1**, **-dtls1_2**

These options specify to use DTLS instead of TLS. With **-dtls**, clients will negotiate any supported DTLS protocol version. Use the **-dtls1** or **-dtls1_2** options to support only DTLS1.0 or DTLS1.2, respectively.

Engine Options

-engine *id*

Load the engine identified by *id* and use all the methods it implements (algorithms, key storage, etc.), unless specified otherwise in the command-specific documentation or it is configured to do so, as described in "Engine Configuration" in **config**(5).

The engine will be used for key ids specified with **-key** and similar options when an option like **-keyform engine** is given.

A special case is the "loader_attic" engine, which is meant just for internal OpenSSL testing purposes and supports loading keys, parameters, certificates, and CRLs from files. When this engine is used, files with such credentials are read via this engine. Using the "file:" schema is optional; a plain file (path) name will do.

Options specifying keys, like **-key** and similar, can use the generic OpenSSL engine key loading URI scheme "org.openssl.engine:" to retrieve private keys and public keys. The URI syntax is as follows, in simplified form:

org.openssl.engine:{engineid}:{keyid}

Where "{engineid}" is the identity/name of the engine, and "{keyid}" is a key identifier that's acceptable by that engine. For example, when using an engine that interfaces against a PKCS#11 implementation, the generic key URI would be something like this (this happens to be an example for the PKCS#11 engine that's part of OpenSC):

-key org.openssl.engine:pkcs11:label_some-private-key

As a third possibility, for engines and providers that have implemented their own **OSSL_STORE_LOADER**(3), "org.openssl.engine:" should not be necessary. For a PKCS#11 implementation that has implemented such a loader, the PKCS#11 URI as defined in RFC 7512 should be possible to use directly:

-key pkcs11:object=some-private-key;pin-value=1234

Provider Options

-provider *name*

Load and initialize the provider identified by *name*. The *name* can be also a path to the provider module. In that case the provider name will be the specified path and not just the provider module name. Interpretation of relative paths is platform specific. The configured "MODULESDIR" path, **OPENSSL_MODULES** environment variable, or the path specified by **-provider-path** is prepended to relative paths. See **provider**(7) for a more detailed description.

-provider-path *path*

Specifies the search path that is to be used for looking for providers. Equivalently, the **OPENSSL_MODULES** environment variable may be set.

-propquery *propq*

Specifies the *property query clause* to be used when fetching algorithms from the loaded providers. See **property**(7) for a more detailed description.

ENVIRONMENT

The OpenSSL library can be take some configuration parameters from the environment. Some of these variables are listed below. For information about specific commands, see **openssl-engine**(1), **openssl-rehash**(1), and **tsget**(1).

For information about the use of environment variables in configuration, see "ENVIRONMENT" in **config**(5).

For information about querying or specifying CPU architecture flags, see **OPENSSL_ia32cap**(3), and **OPENSSL_s390xcap**(3).

For information about all environment variables used by the OpenSSL libraries, see **openssl-env**(7).

OPENSSL_TRACE=*name*[,...]

Enable tracing output of OpenSSL library, by name. This output will only make sense if you know OpenSSL internals well. Also, it might not give you any output at all, depending on how OpenSSL was built.

The value is a comma separated list of names, with the following available:

TRACE

Traces the OpenSSL trace API itself.

INIT

Traces OpenSSL library initialization and cleanup.

TLS

Traces the TLS/SSL protocol.

TLS_CIPHER

Traces the ciphers used by the TLS/SSL protocol.

CONF

Show details about provider and engine configuration.

ENGINE_TABLE

The function that is used by RSA, DSA (etc) code to select registered ENGINEs, cache defaults and functional references (etc), will generate debugging summaries.

ENGINE_REF_COUNT
Reference counts in the ENGINE structure will be monitored with a line of generated for each change.

PKCS5V2
Traces PKCS#5 v2 key generation.

PKCS12_KEYGEN
Traces PKCS#12 key generation.

PKCS12_DECRYPT
Traces PKCS#12 decryption.

X509V3_POLICY
Generates the complete policy tree at various points during X.509 v3 policy evaluation.

BN_CTX
Traces BIGNUM context operations.

CMP
Traces CMP client and server activity.

STORE
Traces STORE operations.

DECODER
Traces decoder operations.

ENCODER
Traces encoder operations.

REF_COUNT
Traces decrementing certain ASN.1 structure references.

SEE ALSO

openssl-asn1parse(1), **openssl-ca**(1), **openssl-ciphers**(1), **openssl-cms**(1), **openssl-crl**(1), **openssl-crl2pkcs7**(1), **openssl-dgst**(1), **openssl-dhparam**(1), **openssl-dsa**(1), **openssl-dsaparam**(1),

openssl-ec(1), **openssl-ecparam**(1), **openssl-enc**(1), **openssl-engine**(1), **openssl-errstr**(1), **openssl-gendsa**(1), **openssl-genpkey**(1), **openssl-genrsa**(1), **openssl-kdf**(1), **openssl-list**(1), **openssl-mac**(1), **openssl-nseq**(1), **openssl-ocsp**(1), **openssl-passwd**(1), **openssl-pkcs12**(1), **openssl-pkcs7**(1), **openssl-pkcs8**(1), **openssl-pkey**(1), **openssl-pkeyparam**(1), **openssl-pkeyutl**(1), **openssl-prime**(1), **openssl-rand**(1), **openssl-rehash**(1), **openssl-req**(1), **openssl-rsa**(1), **openssl-rsautl**(1), **openssl-s_client**(1), **openssl-s_server**(1), **openssl-s_time**(1), **openssl-sess_id**(1), **openssl-smime**(1), **openssl-speed**(1), **openssl-spkac**(1), **openssl-srp**(1), **openssl-storeutl**(1), **openssl-ts**(1), **openssl-verify**(1), **openssl-version**(1), **openssl-x509**(1), **config**(5), **crypto**(7), **openssl-env**(7). **ssl**(7), **x509v3_config**(5)

HISTORY

The **list** -*XXX*-**algorithms** options were added in OpenSSL 1.0.0; For notes on the availability of other commands, see their individual manual pages.

The **-issuer_checks** option is deprecated as of OpenSSL 1.1.0 and is silently ignored.

The **-xcertform** and **-xkeyform** options are obsolete since OpenSSL 3.0 and have no effect.

The interactive mode, which could be invoked by running "openssl" with no further arguments, was removed in OpenSSL 3.0, and running that program with no arguments is now equivalent to "openssl help".

COPYRIGHT