
## *Rocky Enterprise Linux 9.2 Manual Pages on command 'openssl-dgst.1ssl'*

***C:\>man openssl-dgst.1ssl***

DGST(1SSL) OpenSSL DGST(1SSL)

NAME

openssl-dgst, dgst - perform digest operations

SYNOPSIS

openssl dgst [-digest] [-help] [-c] [-d] [-list] [-hex] [-binary] [-r] [-out filename] [-sign filename] [-keyform arg] [-passin arg] [-verify filename] [-prverify filename] [-signature filename] [-sigopt nm:v] [-hmac key] [-fips-fingerprint] [-rand file...] [-engine id] [-engine_impl] [file...]

openssl digest [...]

DESCRIPTION

The digest functions output the message digest of a supplied file or files in hexadecimal. The digest functions also generate and verify digital signatures using message digests.

The generic name, dgst, may be used with an option specifying the algorithm to be used. The default digest is sha256. A supported digest name may also be used as the command name. To see the list of supported algorithms, use the list --digest-commands command.

OPTIONS

-help

Print out a usage message.

-digest

Specifies name of a supported digest to be used. To see the list of supported

digests, use the command list --digest-commands.

-c Print out the digest in two digit groups separated by colons, only relevant if hex format output is used.

-d Print out BIO debugging information.

-list

Prints out a list of supported message digests.

-hex

Digest is to be output as a hex dump. This is the default case for a "normal" digest as opposed to a digital signature. See NOTES below for digital signatures using -hex.

-binary

Output the digest or signature in binary form.

-r Output the digest in the "coreutils" format, including newlines. Used by programs like sha1sum.

-out filename

Filename to output to, or standard output by default.

-sign filename

Digitally sign the digest using the private key in "filename". Note this option does not support Ed25519 or Ed448 private keys. Use the pkeyutl command instead for this.

-keyform arg

Specifies the key format to sign digest with. The DER, PEM, P12, and ENGINE formats are supported.

-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 private key password source. For more information about the format of arg see the PASS PHRASE ARGUMENTS section in openssl(1).

-verify filename

Verify the signature using the public key in "filename". The output is either "Verification OK" or "Verification Failure".

-prverify filename

Verify the signature using the private key in "filename".

-signature filename

The actual signature to verify.

-hmac key

Create a hashed MAC using "key".

-mac alg

Create MAC (keyed Message Authentication Code). The most popular MAC algorithm is HMAC (hash-based MAC), but there are other MAC algorithms which are not based on hash, for instance gost-mac algorithm, supported by ccgost engine. MAC keys and other options should be set via -macopt parameter.

-macopt nm:v

Passes options to MAC algorithm, specified by -mac key. Following options are supported by both by HMAC and gost-mac:

key:string

Specifies MAC key as alphanumeric string (use if key contain printable characters only). String length must conform to any restrictions of the MAC algorithm for example exactly 32 chars for gost-mac.

hexkey:string

Specifies MAC key in hexadecimal form (two hex digits per byte). Key length must conform to any restrictions of the MAC algorithm for example exactly 32 chars for gost-mac.

-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.

-fips-fingerprint

Compute HMAC using a specific key for certain OpenSSL-FIPS operations.

-engine id

Use engine id for operations (including private key storage). This engine is

not used as source for digest algorithms, unless it is also specified in the configuration file or -engine_impl is also specified.

-engine_impl

When used with the -engine option, it specifies to also use engine id for digest operations.

file...

File or files to digest. If no files are specified then standard input is used.

EXAMPLES

To create a hex-encoded message digest of a file:

```
openssl dgst -md5 -hex file.txt
```

To sign a file using SHA-256 with binary file output:

```
openssl dgst -sha256 -sign privatekey.pem -out signature.sign file.txt
```

To verify a signature:

```
openssl dgst -sha256 -verify publickey.pem \
-signature signature.sign \
file.txt
```

NOTES

The digest mechanisms that are available will depend on the options used when building OpenSSL. The list digest-commands command can be used to list them.

New or agile applications should use probably use SHA-256. Other digests, particularly SHA-1 and MD5, are still widely used for interoperating with existing formats and protocols.

When signing a file, dgst will automatically determine the algorithm (RSA, ECC, etc) to use for signing based on the private key's ASN.1 info. When verifying signatures, it only handles the RSA, DSA, or ECDSA signature itself, not the related data to identify the signer and algorithm used in formats such as x.509, CMS, and S/MIME.

A source of random numbers is required for certain signing algorithms, in particular ECDSA and DSA.

The signing and verify options should only be used if a single file is being signed or verified.

Hex signatures cannot be verified using openssl. Instead, use "xxd -r" or similar program to transform the hex signature into a binary signature prior to

verification.

## HISTORY

The default digest was changed from MD5 to SHA256 in OpenSSL 1.1.0. The FIPS-related options were removed in OpenSSL 1.1.0.

## COPYRIGHT