
*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 'CMS_sign.3ossl' command*

***$ man CMS_sign.3ossl***

CMS_SIGN(3ossl) OpenSSL CMS_SIGN(3ossl)

NAME

CMS_sign, CMS_sign_ex - create a CMS SignedData structure

SYNOPSIS

```
#include <openssl/cms.h>
CMS_ContentInfo *CMS_sign_ex(X509 *signcert, EVP_PKEY *pkey,
                             STACK_OF(X509) *certs, BIO *data,
                             unsigned int flags, OSSL_LIB_CTX *ctx,
                             const char *propq);
CMS_ContentInfo *CMS_sign(X509 *signcert, EVP_PKEY *pkey, STACK_OF(X509) *certs,
                          BIO *data, unsigned int flags);
```

DESCRIPTION

CMS_sign_ex() creates and returns a CMS SignedData structure. signcert is the certificate to sign with, pkey is the corresponding private key. certs is an optional additional set of certificates to include in the CMS structure (for example any intermediate CAs in the chain). The library context libctx and the property query propq are used when retrieving algorithms from providers. Any or all of these parameters can be NULL, see NOTES below.

The data to be signed is read from BIO data.

flags is an optional set of flags.

CMS_sign() is similar to CMS_sign_ex() but uses default values of NULL for the library context libctx and the property query propq.

## NOTES

Any of the following flags (ored together) can be passed in the flags parameter.

Many S/MIME clients expect the signed content to include valid MIME headers. If the CMS_TEXT flag is set MIME headers for type text/plain are prepended to the data.

If CMS_NOCERTS is set the signer's certificate will not be included in the CMS_ContentInfo structure, the signer's certificate must still be supplied in the signcert parameter though. This can reduce the size of the signature if the signers certificate can be obtained by other means: for example a previously signed message.

The data being signed is included in the CMS_ContentInfo structure, unless CMS_DETACHED is set in which case it is omitted. This is used for CMS_ContentInfo detached signatures which are used in S/MIME plaintext signed messages for example.

Normally the supplied content is translated into MIME canonical format (as required by the S/MIME specifications) if CMS_BINARY is set no translation occurs. This option should be used if the supplied data is in binary format otherwise the translation will corrupt it.

The SignedData structure includes several CMS signedAttributes including the signing time, the CMS content type and the supported list of ciphers in an SMIMECapabilities attribute. If CMS_NOATTR is set then no signedAttributes will be used. If CMS_NOSMIMECAP is set then just the SMIMECapabilities are omitted.

If present the SMIMECapabilities attribute indicates support for the following algorithms in preference order: 256 bit AES, Gost R3411-94, Gost 28147-89, 192 bit AES, 128 bit AES, triple DES, 128 bit RC2, 64 bit RC2, DES and 40 bit RC2.  If any of these algorithms is not available then it will not be included: for example the GOST algorithms will not be included if the GOST ENGINE is not loaded.

OpenSSL will by default identify signing certificates using issuer name and serial number. If CMS_USE_KEYID is set it will use the subject key identifier value instead. An error occurs if the signing certificate

does not have a subject key identifier extension.

If the flags CMS_STREAM is set then the returned CMS_ContentInfo structure is just initialized ready to perform the signing operation. The signing is however not performed and the data to be signed is not read from the data parameter. Signing is deferred until after the data has been written. In this way data can be signed in a single pass.

If the CMS_PARTIAL flag is set a partial CMS_ContentInfo structure is output to which additional signers and capabilities can be added before finalization.

If the flag CMS_STREAM is set the returned CMS_ContentInfo structure is not complete and outputting its contents via a function that does not properly finalize the CMS_ContentInfo structure will give unpredictable results.

Several functions including SMIME_write_CMS(), i2d_CMS_bio_stream(), PEM_write_bio_CMS_stream() finalize the structure. Alternatively finalization can be performed by obtaining the streaming ASN1 BIO directly using BIO_new_CMS().

If a signer is specified it will use the default digest for the signing algorithm. This is SHA1 for both RSA and DSA keys.

If signcert and pkey are NULL then a certificates only CMS structure is output.

The function CMS_sign() is a basic CMS signing function whose output will be suitable for many purposes. For finer control of the output format the certs, signcert and pkey parameters can all be NULL and the CMS_PARTIAL flag set. Then one or more signers can be added using the function CMS_sign_add1_signer(), non default digests can be used and custom attributes added. CMS_final() must then be called to finalize the structure if streaming is not enabled.

## BUGS

Some attributes such as counter signatures are not supported.

## RETURN VALUES

CMS_sign_ex() and CMS_sign() return either a valid CMS_ContentInfo structure or NULL if an error occurred. The error can be obtained from

ERR_get_error(3).

SEE ALSO

ERR_get_error(3), CMS_verify(3)

HISTORY

The CMS_STREAM flag is only supported for detached data in OpenSSL 0.9.8, it is supported for embedded data in OpenSSL 1.0.0 and later.

The CMS_sign_ex() method was added in OpenSSL 3.0.

COPYRIGHT