NAME

CMS_add1_signer, CMS_SignerInfo_sign - add a signer to a CMS_ContentInfo signed data structure

SYNOPSIS

```
 #include <openssl/cms.h>

 CMS_SignerInfo *CMS_add1_signer(CMS_ContentInfo *cms, X509 *signcert,
                                 EVP_PKEY *pkey, const EVP_MD *md,
                                 unsigned int flags);

 int CMS_SignerInfo_sign(CMS_SignerInfo *si);
```

DESCRIPTION

CMS_add1_signer() adds a signer with certificate **signcert** and private key **pkey** using message digest **md** to CMS_ContentInfo SignedData structure **cms**.

The CMS_ContentInfo structure should be obtained from an initial call to **CMS_sign()** with the flag **CMS_PARTIAL** set or in the case or re-signing a valid CMS_ContentInfo SignedData structure.

If the **md** parameter is **NULL** then the default digest for the public key algorithm will be used.

Unless the **CMS_REUSE_DIGEST** flag is set the returned CMS_ContentInfo structure is not complete and must be finalized either by streaming (if applicable) or a call to **CMS_final()**.

The **CMS_SignerInfo_sign()** function will explicitly sign a CMS_SignerInfo structure, its main use is when **CMS_REUSE_DIGEST** and **CMS_PARTIAL** flags are both set.

NOTES

The main purpose of **CMS_add1_signer()** is to provide finer control over a CMS signed data structure where the simpler **CMS_sign()** function defaults are not appropriate. For example if multiple signers or non default digest algorithms are needed. New attributes can also be added using the returned CMS_SignerInfo structure and the CMS attribute utility functions or the CMS signed receipt request functions.

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

If **CMS_REUSE_DIGEST** is set then an attempt is made to copy the content digest value from the CMS_ContentInfo structure: to add a signer to an existing structure. An error occurs if a matching digest value cannot be found to copy. The returned CMS_ContentInfo structure will be valid and finalized when this flag is set.

If **CMS_PARTIAL** is set in addition to **CMS_REUSE_DIGEST** then the CMS_SignerInfo structure will not be finalized so additional attributes can be added. In this case an explicit call to **CMS_SignerInfo_sign()** is needed to finalize it.

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 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.

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 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.

CMS_add1_signer() returns an internal pointer to the CMS_SignerInfo structure just added, this can be used to set additional attributes before it is finalized.

RETURN VALUES

CMS_add1_signer() returns an internal pointer to the CMS_SignerInfo structure just added or NULL if an error occurs.

SEE ALSO

ERR_get_error(3), **CMS_sign**(3), **CMS_final**(3),

COPYRIGHT