

**NAME**

CMS\_get0\_RecipientInfos, CMS\_RecipientInfo\_type,  
 CMS\_RecipientInfo\_ktri\_get0\_signer\_id, CMS\_RecipientInfo\_ktri\_cert\_cmp,  
 CMS\_RecipientInfo\_set0\_pkey, CMS\_RecipientInfo\_kekri\_get0\_id,  
 CMS\_RecipientInfo\_kekri\_id\_cmp, CMS\_RecipientInfo\_set0\_key, CMS\_RecipientInfo\_decrypt,  
 CMS\_RecipientInfo\_encrypt - CMS envelopedData RecipientInfo routines

**SYNOPSIS**

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

```
STACK_OF(CMS_RecipientInfo) *CMS_get0_RecipientInfos(CMS_ContentInfo *cms);
int CMS_RecipientInfo_type(CMS_RecipientInfo *ri);
```

```
int CMS_RecipientInfo_ktri_get0_signer_id(CMS_RecipientInfo *ri, ASN1_OCTET_STRING **keyid,
int CMS_RecipientInfo_ktri_cert_cmp(CMS_RecipientInfo *ri, X509 *cert);
int CMS_RecipientInfo_set0_pkey(CMS_RecipientInfo *ri, EVP_PKEY *pkey);
```

```
int CMS_RecipientInfo_kekri_get0_id(CMS_RecipientInfo *ri, X509_ALGOR **palg, ASN1_OCTET_STRING **id,
int CMS_RecipientInfo_kekri_id_cmp(CMS_RecipientInfo *ri, const unsigned char *id, size_t idlen);
int CMS_RecipientInfo_set0_key(CMS_RecipientInfo *ri, unsigned char *key, size_t keylen);
```

```
int CMS_RecipientInfo_decrypt(CMS_ContentInfo *cms, CMS_RecipientInfo *ri);
int CMS_RecipientInfo_encrypt(CMS_ContentInfo *cms, CMS_RecipientInfo *ri);
```

**DESCRIPTION**

The function *CMS\_get0\_RecipientInfos()* returns all the CMS\_RecipientInfo structures associated with a CMS EnvelopedData structure.

*CMS\_RecipientInfo\_type()* returns the type of CMS\_RecipientInfo structure **ri**. It will currently return CMS\_RECIPINFO\_TRANS, CMS\_RECIPINFO\_AGREE, CMS\_RECIPINFO\_KEK, CMS\_RECIPINFO\_PASS, or CMS\_RECIPINFO\_OTHER.

*CMS\_RecipientInfo\_ktri\_get0\_signer\_id()* retrieves the certificate recipient identifier associated with a specific CMS\_RecipientInfo structure **ri**, which must be of type CMS\_RECIPINFO\_TRANS. Either the keyIdentifier will be set in **keyid** or **both** issuer name and serial number in **issuer** and **sno**.

*CMS\_RecipientInfo\_ktri\_cert\_cmp()* compares the certificate **cert** against the CMS\_RecipientInfo structure **ri**, which must be of type CMS\_RECIPINFO\_TRANS. It returns zero if the comparison is successful and non zero if not.

*CMS\_RecipientInfo\_set0\_pkey()* associates the private key **pkey** with the CMS\_RecipientInfo structure **ri**, which must be of type CMS\_RECIPINFO\_TRANS.

*CMS\_RecipientInfo\_kekri\_get0\_id()* retrieves the key information from the CMS\_RecipientInfo structure **ri** which must be of type CMS\_RECIPINFO\_KEK. Any of the remaining parameters can be NULL if the application is not interested in the value of a field. Where a field is optional and absent NULL will be written to the corresponding parameter. The keyEncryptionAlgorithm field is written to **palg**, the **keyIdentifier** field is written to **pid**, the **date** field if present is written to **pdate**, if the **other** field is present the components **keyAttrId** and **keyAttr** are written to parameters **pothetid** and **pothertype**.

*CMS\_RecipientInfo\_kekri\_id\_cmp()* compares the ID in the **id** and **idlen** parameters against the **keyIdentifier** CMS\_RecipientInfo structure **ri**, which must be of type CMS\_RECIPINFO\_KEK. It returns zero if the comparison is successful and non zero if not.

*CMS\_RecipientInfo\_set0\_key()* associates the symmetric key **key** of length **keylen** with the CMS\_RecipientInfo structure **ri**, which must be of type CMS\_RECIPINFO\_KEK.

*CMS\_RecipientInfo\_decrypt()* attempts to decrypt CMS\_RecipientInfo structure **ri** in structure

**cms.** A key must have been associated with the structure first.

*CMS\_RecipientInfo\_encrypt()* attempts to encrypt CMS\_RecipientInfo structure **ri** in structure **cms**. A key must have been associated with the structure first and the content encryption key must be available: for example by a previous call to *CMS\_RecipientInfo\_decrypt()*.

## NOTES

The main purpose of these functions is to enable an application to lookup recipient keys using any appropriate technique when the simpler method of *CMS\_decrypt()* is not appropriate.

In typical usage and application will retrieve all CMS\_RecipientInfo structures using *CMS\_get0\_RecipientInfos()* and check the type of each using *CMS\_RecipientInfo\_type()*. Depending on the type the CMS\_RecipientInfo structure can be ignored or its key identifier data retrieved using an appropriate function. Then if the corresponding secret or private key can be obtained by any appropriate means it can then associated with the structure and *CMS\_RecipientInfo\_decrypt()* called. If successful *CMS\_decrypt()* can be called with a NULL key to decrypt the enveloped content.

The *CMS\_RecipientInfo\_encrypt()* can be used to add a new recipient to an existing enveloped data structure. Typically an application will first decrypt an appropriate CMS\_RecipientInfo structure to make the content encrypt key available, it will then add a new recipient using a function such as *CMS\_add1\_recipient\_cert()* and finally encrypt the content encryption key using *CMS\_RecipientInfo\_encrypt()*.

## RETURN VALUES

*CMS\_get0\_RecipientInfos()* returns all CMS\_RecipientInfo structures, or NULL if an error occurs.

*CMS\_RecipientInfo\_ktri\_get0\_signer\_id()*, *CMS\_RecipientInfo\_set0\_pkey()*,  
*CMS\_RecipientInfo\_kekri\_get0\_id()*, *CMS\_RecipientInfo\_set0\_key()* and  
*CMS\_RecipientInfo\_decrypt()* return 1 for success or 0 if an error occurs.  
*CMS\_RecipientInfo\_encrypt()* return 1 for success or 0 if an error occurs.

*CMS\_RecipientInfo\_ktri\_cert\_cmp()* and *CMS\_RecipientInfo\_kekri\_cmp()* return 0 for a successful comparison and non zero otherwise.

Any error can be obtained from [ERR\\_get\\_error\(3\)](#).

## SEE ALSO

[ERR\\_get\\_error\(3\)](#), [CMS\\_decrypt\(3\)](#)

## HISTORY

These functions were first added to OpenSSL 0.9.8