NAME

PKCS7_verify, PKCS7_get0_signers - verify a PKCS#7 signedData structure

SYNOPSIS

```
 #include <openssl/pkcs7.h>

 int PKCS7_verify(PKCS7 *p7, STACK_OF(X509) *certs, X509_STORE *store,
                  BIO *indata, BIO *out, int flags);

 STACK_OF(X509) *PKCS7_get0_signers(PKCS7 *p7, STACK_OF(X509) *certs, int flags);
```

DESCRIPTION

PKCS7_verify() is very similar to **CMS_verify**(3). It verifies a PKCS#7 signedData structure given in *p7*. The optional *certs* parameter refers to a set of certificates in which to search for signer's certificates. *p7* may contain extra untrusted CA certificates that may be used for chain building as well as CRLs that may be used for certificate validation. *store* may be NULL or point to the trusted certificate store to use for chain verification. *indata* refers to the signed data if the content is detached from *p7*. Otherwise *indata* should be NULL, and then the signed data must be in *p7*. The content is written to the BIO *out* unless it is NULL. *flags* is an optional set of flags, which can be used to modify the operation.

PKCS7_get0_signers() retrieves the signer's certificates from *p7*, it does **not** check their validity or whether any signatures are valid. The *certs* and *flags* parameters have the same meanings as in **PKCS7_verify()**.

VERIFY PROCESS

Normally the verify process proceeds as follows.

Initially some sanity checks are performed on *p7*. The type of *p7* must be SignedData. There must be at least one signature on the data and if the content is detached *indata* cannot be NULL. If the content is not detached and *indata* is not NULL then the structure has both embedded and external content. To treat this as an error, use the flag **PKCS7_NO_DUAL_CONTENT**. The default behavior allows this, for compatibility with older versions of OpenSSL.

An attempt is made to locate all the signer's certificates, first looking in the *certs* parameter (if it is not NULL). Then they are looked up in any certificates contained in the *p7* structure unless **PKCS7_NOINTERN** is set. If any signer's certificates cannot be located the operation fails.

Each signer's certificate is chain verified using the **smimesign** purpose and using the trusted certificate store *store* if supplied. Any internal certificates in the message, which may have been added using

PKCS7_add_certificate(3), are used as untrusted CAs unless **PKCS7_NOCHAIN** is set. If CRL checking is enabled in *store* and **PKCS7_NOCRL** is not set, any internal CRLs, which may have been added using **PKCS7_add_crl**(3), are used in addition to attempting to look them up in *store*. If *store* is not NULL and any chain verify fails an error code is returned.

Finally the signed content is read (and written to *out* unless it is NULL) and the signature is checked.

If all signatures verify correctly then the function is successful.

Any of the following flags (ored together) can be passed in the *flags* parameter to change the default verify behaviour. Only the flag **PKCS7_NOINTERN** is meaningful to **PKCS7_get0_signers()**.

If **PKCS7_NOINTERN** is set the certificates in the message itself are not searched when locating the signer's certificates. This means that all the signer's certificates must be in the *certs* parameter.

If **PKCS7_NOCRL** is set and CRL checking is enabled in *store* then any CRLs in the message itself are ignored.

If the **PKCS7_TEXT** flag is set MIME headers for type "text/plain" are deleted from the content. If the content is not of type "text/plain" then an error is returned.

If **PKCS7_NOVERIFY** is set the signer's certificates are not chain verified.

If **PKCS7_NOCHAIN** is set then the certificates contained in the message are not used as untrusted CAs. This means that the whole verify chain (apart from the signer's certificates) must be contained in the trusted store.

If **PKCS7_NOSIGS** is set then the signatures on the data are not checked.

NOTES

One application of **PKCS7_NOINTERN** is to only accept messages signed by a small number of certificates. The acceptable certificates would be passed in the *certs* parameter. In this case if the signer's certificate is not one of the certificates supplied in *certs* then the verify will fail because the signer cannot be found.

Care should be taken when modifying the default verify behaviour, for example setting "PKCS7_NOVERIFY|PKCS7_NOSIGS" will totally disable all verification and any signed message will be considered valid. This combination is however useful if one merely wishes to write the content to *out* and its validity is not considered important.

Chain verification should arguably be performed using the signing time rather than the current time. However, since the signing time is supplied by the signer it cannot be trusted without additional evidence (such as a trusted timestamp).

RETURN VALUES

PKCS7_verify() returns 1 for a successful verification and 0 if an error occurs.

PKCS7_get0_signers() returns all signers or NULL if an error occurred.

The error can be obtained from **ERR_get_error**(3).

BUGS

The trusted certificate store is not searched for the signer's certificates. This is primarily due to the inadequacies of the current **X509_STORE** functionality.

The lack of single pass processing means that the signed content must all be held in memory if it is not detached.

SEE ALSO

CMS_verify(3), **PKCS7_add_certificate**(3), **PKCS7_add_crl**(3), **ERR_get_error**(3), **PKCS7_sign**(3)

COPYRIGHT

Copyright 2002-2022 The OpenSSL Project Authors. All Rights Reserved.

Licensed under the Apache License 2.0 (the "License"). You may not use this file except in compliance with the License. You can obtain a copy in the file LICENSE in the source distribution or at <https://www.openssl.org/source/license.html>.