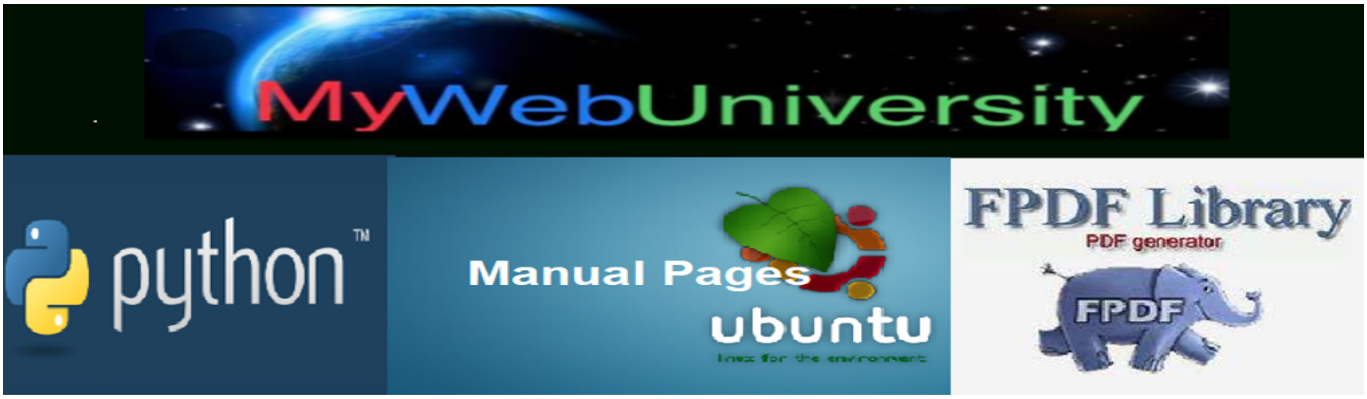

Full credit is given to the above companies including the Operating System (OS) that this PDF file was generated!

Rocky Enterprise Linux 9.2 Manual Pages on command 'openssl-smime.1ssl'

$ man openssl-smime.1ssl

OPENSSL-SMIME(1SSL) OpenSSL OPENSSL-SMIME(1SSL)

NAME

openssl-smime - S/MIME command

SYNOPSIS

openssl smime [-help] [-encrypt] [-decrypt] [-sign] [-resign] [-verify] [-pk7out]

[-binary] [-crlfeol] [-cipher] [-in file] [-certfile file] [-signer file] [-nointern]

[-noverify] [-nochain] [-nosigs] [-nocerts] [-noattr] [-nodetach] [-nosmimecap] [-recip

file] [-inform DER|PEM|SMIME] [-outform DER|PEM|SMIME] [-keyform DER|PEM|P12|ENGINE]

[-passin arg] [-inkey filename|uri] [-out file] [-content file] [-to addr] [-from ad]

[-subject s] [-text] [-indef] [-noindef] [-stream] [-md digest] [-CAfile file]

[-no-CAfile] [-CApath dir] [-no-CApath] [-CAstore uri] [-no-CAstore] [-engine id] [-rand

files] [-writerand file] [-allow_proxy_certs] [-attime timestamp] [-no_check_time]

[-check_ss_sig] [-crl_check] [-crl_check_all] [-explicit_policy] [-extended_crl]

[-ignore_critical] [-inhibit_any] [-inhibit_map] [-partial_chain] [-policy arg]

[-policy_check] [-policy_print] [-purpose purpose] [-suiteB_128] [-suiteB_128_only]

[-suiteB_192] [-trusted_first] [-no_alt_chains] [-use_deltas] [-auth_level num]

[-verify_depth num] [-verify_email email] [-verify_hostname hostname] [-verify_ip ip]

[-verify_name name] [-x509_strict] [-issuer_checks] [-provider name] [-provider-path path]

[-propquery propq] [-config configfile] recipcert ...

DESCRIPTION

This command handles S/MIME mail. It can encrypt, decrypt, sign and verify S/MIME

messages.

OPTIONS

There are six operation options that set the type of operation to be performed. The meaning of the other options varies according to the operation type.

-help

Print out a usage message.

-encrypt

Encrypt mail for the given recipient certificates. Input file is the message to be encrypted. The output file is the encrypted mail in MIME format.

Note that no revocation check is done for the recipient cert, so if that key has been compromised, others may be able to decrypt the text.

-decrypt

Decrypt mail using the supplied certificate and private key. Expects an encrypted mail message in MIME format for the input file. The decrypted mail is written to the output file.

-sign

Sign mail using the supplied certificate and private key. Input file is the message to be signed. The signed message in MIME format is written to the output file.

-verify

Verify signed mail. Expects a signed mail message on input and outputs the signed data. Both clear text and opaque signing is supported.

-pk7out

Takes an input message and writes out a PEM encoded PKCS#7 structure.

-resign

Resign a message: take an existing message and one or more new signers.

-in filename

The input message to be encrypted or signed or the MIME message to be decrypted or verified.

-out filename

The message text that has been decrypted or verified or the output MIME format message that has been signed or verified.

-inform DER|PEM|SMIME

The input format of the PKCS#7 (S/MIME) structure (if one is being read); the default is SMIME. See openssl-format-options(1) for details.

-outform DER|PEM|SMIME

The output format of the PKCS#7 (S/MIME) structure (if one is being written); the default is SMIME. See openssl-format-options(1) for details.

-keyform DER|PEM|P12|ENGINE

The key format; unspecified by default. See openssl-format-options(1) for details.

-stream, -indef, -noindef

The -stream and -indef options are equivalent and enable streaming I/O for encoding operations. This permits single pass processing of data without the need to hold the entire contents in memory, potentially supporting very large files. Streaming is automatically set for S/MIME signing with detached data if the output format is SMIME it is currently off by default for all other operations.

-noindef

Disable streaming I/O where it would produce and indefinite length constructed encoding. This option currently has no effect. In future streaming will be enabled by default on all relevant operations and this option will disable it.

-content filename

This specifies a file containing the detached content, this is only useful with the -verify command. This is only usable if the PKCS#7 structure is using the detached signature form where the content is not included. This option will override any content if the input format is S/MIME and it uses the multipart/signed MIME content type.

-text

This option adds plain text (text/plain) MIME headers to the supplied message if encrypting or signing. If decrypting or verifying it strips off text headers: if the decrypted or verified message is not of MIME type text/plain then an error occurs.

-md digest

Digest algorithm to use when signing or resigning. If not present then the default digest algorithm for the signing key will be used (usually SHA1).

-cipher

The encryption algorithm to use. For example DES (56 bits) - -des, triple DES (168 bits) - -des3, EVP_get_cipherbyname() function) can also be used preceded by a dash, for example -aes-128-cbc. See openssl-enc(1) for list of ciphers supported by your version of OpenSSL.

If not specified triple DES is used. Only used with -encrypt.

-nointern

When verifying a message normally certificates (if any) included in the message are searched for the signing certificate. With this option only the certificates specified in the -certfile option are used. The supplied certificates can still be used as untrusted CAs however.

-noverify

Do not verify the signers certificate of a signed message.

-nochain

Do not do chain verification of signers certificates; that is, do not use the certificates in the signed message as untrusted CAs.

-nosigs

Don't try to verify the signatures on the message.

-nocerts

When signing a message the signer's certificate is normally included with this option it is excluded. This will reduce the size of the signed message but the verifier must have a copy of the signers certificate available locally (passed using the -certfile option for example).

-noattr

Normally when a message is signed a set of attributes are included which include the signing time and supported symmetric algorithms. With this option they are not included.

-nodetach

When signing a message use opaque signing. This form is more resistant to translation by mail relays but it cannot be read by mail agents that do not support S/MIME. Without this option cleartext signing with the MIME type multipart/signed is used.

-nosmimecap

When signing a message, do not include the SMIMECapabilities attribute.

-binary

Normally the input message is converted to "canonical" format which is effectively using CR and LF as end of line: as required by the S/MIME specification. When this option is present no translation occurs. This is useful when handling binary data which may not be in MIME format.

-crlfeol

Normally the output file uses a single LF as end of line. When this option is present CRLF is used instead.

-certfile file

Allows additional certificates to be specified. When signing these will be included with the message. When verifying these will be searched for the signers certificates. The input can be in PEM, DER, or PKCS#12 format.

-signer file

A signing certificate when signing or resigning a message, this option can be used multiple times if more than one signer is required. If a message is being verified then the signers certificates will be written to this file if the verification was successful.

-nocerts

Don't include signers certificate when signing.

-noattr

Don't include any signed attributes when signing.

-recip file

The recipients certificate when decrypting a message. This certificate must match one of the recipients of the message or an error occurs.

-inkey filename|uri

The private key to use when signing or decrypting. This must match the corresponding certificate. If this option is not specified then the private key must be included in the certificate file specified with the -recip or -signer file. When signing this option can be used multiple times to specify successive keys.

-passin arg

The private key password source. For more information about the format of arg see openssl-passphrase-options(1).

-to, -from, -subject

The relevant mail headers. These are included outside the signed portion of a message so they may be included manually. If signing then many S/MIME mail clients check the signers certificate's email address matches that specified in the From: address.

-allow_proxy_certs, -attime, -no_check_time, -check_ss_sig, -crl_check, -crl_check_all, -explicit_policy, -extended_crl, -ignore_critical, -inhibit_any, -inhibit_map, -no_alt_chains, -partial_chain, -policy, -policy_check, -policy_print, -purpose,

-suiteB_128, -suiteB_128_only, -suiteB_192, -trusted_first, -use_deltas, -auth_level, -verify_depth, -verify_email, -verify_hostname, -verify_ip, -verify_name, -x509_strict -issuer_checks

Set various options of certificate chain verification. See "Verification Options" in openssl-verification-options(1) for details.

Any verification errors cause the command to exit.

-CAfile file, -no-CAfile, -CApath dir, -no-CApath, -CAstore uri, -no-CAstore

See "Trusted Certificate Options" in openssl-verification-options(1) for details.

-engine id

See "Engine Options" in openssl(1). This option is deprecated.

-rand files, -writerand file

See "Random State Options" in openssl(1) for details.

-provider name

-provider-path path

-propquery propq

See "Provider Options" in openssl(1), provider(7), and property(7).

-config configfile

See "Configuration Option" in openssl(1).

recipcert ...

One or more certificates of message recipients, used when encrypting a message.

NOTES

The MIME message must be sent without any blank lines between the headers and the output. Some mail programs will automatically add a blank line. Piping the mail directly to sendmail is one way to achieve the correct format.

The supplied message to be signed or encrypted must include the necessary MIME headers or many S/MIME clients won't display it properly (if at all). You can use the -text option to automatically add plain text headers.

A "signed and encrypted" message is one where a signed message is then encrypted. This can be produced by encrypting an already signed message: see the examples section.

This version of the program only allows one signer per message but it will verify multiple signers on received messages. Some S/MIME clients choke if a message contains multiple signers. It is possible to sign messages "in parallel" by signing an already signed message.

The options -encrypt and -decrypt reflect common usage in S/MIME clients. Strictly speaking these process PKCS#7 enveloped data: PKCS#7 encrypted data is used for other purposes.

The -resign option uses an existing message digest when adding a new signer. This means that attributes must be present in at least one existing signer using the same message digest or this operation will fail.

The -stream and -indef options enable streaming I/O support. As a result the encoding is BER using indefinite length constructed encoding and no longer DER. Streaming is supported for the -encrypt operation and the -sign operation if the content is not detached.

Streaming is always used for the -sign operation with detached data but since the content is no longer part of the PKCS#7 structure the encoding remains DER.

EXIT CODES

0 The operation was completely successfully.

1 An error occurred parsing the command options.

2 One of the input files could not be read.

3 An error occurred creating the PKCS#7 file or when reading the MIME message.

4 An error occurred decrypting or verifying the message.

5 The message was verified correctly but an error occurred writing out the signers certificates.

EXAMPLES

Create a cleartext signed message:

```
openssl smime -sign -in message.txt -text -out mail.msg \
    -signer mycert.pem
```

Create an opaque signed message:

```
openssl smime -sign -in message.txt -text -out mail.msg -nodetach \
    -signer mycert.pem
```

Create a signed message, include some additional certificates and read the private key from another file:

```
openssl smime -sign -in in.txt -text -out mail.msg \
    -signer mycert.pem -inkey mykey.pem -certfile mycerts.pem
```

Create a signed message with two signers:

```
openssl smime -sign -in message.txt -text -out mail.msg \
    -signer mycert.pem -signer othercert.pem
```

Send a signed message under Unix directly to sendmail, including headers:

```
openssl smime -sign -in in.txt -text -signer mycert.pem \
      -from steve@openssl.org -to someone@somewhere \
      -subject "Signed message" | sendmail someone@somewhere
```

Verify a message and extract the signer's certificate if successful:

```
openssl smime -verify -in mail.msg -signer user.pem -out signedtext.txt
```

Send encrypted mail using triple DES:

```
openssl smime -encrypt -in in.txt -from steve@openssl.org \
      -to someone@somewhere -subject "Encrypted message" \
      -des3 user.pem -out mail.msg
```

Sign and encrypt mail:

```
openssl smime -sign -in ml.txt -signer my.pem -text \
      | openssl smime -encrypt -out mail.msg \
      -from steve@openssl.org -to someone@somewhere \
      -subject "Signed and Encrypted message" -des3 user.pem
```

Note: the encryption command does not include the -text option because the message being encrypted already has MIME headers.

Decrypt mail:

```
openssl smime -decrypt -in mail.msg -recip mycert.pem -inkey key.pem
```

The output from Netscape form signing is a PKCS#7 structure with the detached signature format. You can use this program to verify the signature by line wrapping the base64 encoded structure and surrounding it with:

```
-----BEGIN PKCS7-----
-----END PKCS7-----
```

and using the command:

```
openssl smime -verify -inform PEM -in signature.pem -content content.txt
```

Alternatively you can base64 decode the signature and use:

```
openssl smime -verify -inform DER -in signature.der -content content.txt
```

Create an encrypted message using 128 bit Camellia:

```
openssl smime -encrypt -in plain.txt -camellia128 -out mail.msg cert.pem
```

Add a signer to an existing message:

```
openssl smime -resign -in mail.msg -signer newsign.pem -out mail2.msg
```

BUGS

The MIME parser isn't very clever: it seems to handle most messages that I've thrown at it but it may choke on others.

The code currently will only write out the signer's certificate to a file: if the signer has a separate encryption certificate this must be manually extracted. There should be some heuristic that determines the correct encryption certificate.

Ideally a database should be maintained of a certificates for each email address.

The code doesn't currently take note of the permitted symmetric encryption algorithms as supplied in the SMIMECapabilities signed attribute. This means the user has to manually include the correct encryption algorithm. It should store the list of permitted ciphers in a database and only use those.

No revocation checking is done on the signer's certificate.

The current code can only handle S/MIME v2 messages, the more complex S/MIME v3 structures may cause parsing errors.

SEE ALSO

ossl_store-file(7)

HISTORY

The use of multiple -signer options and the -resign command were first added in OpenSSL 1.0.0

The -no_alt_chains option was added in OpenSSL 1.1.0.

The -engine option was deprecated in OpenSSL 3.0.

COPYRIGHT