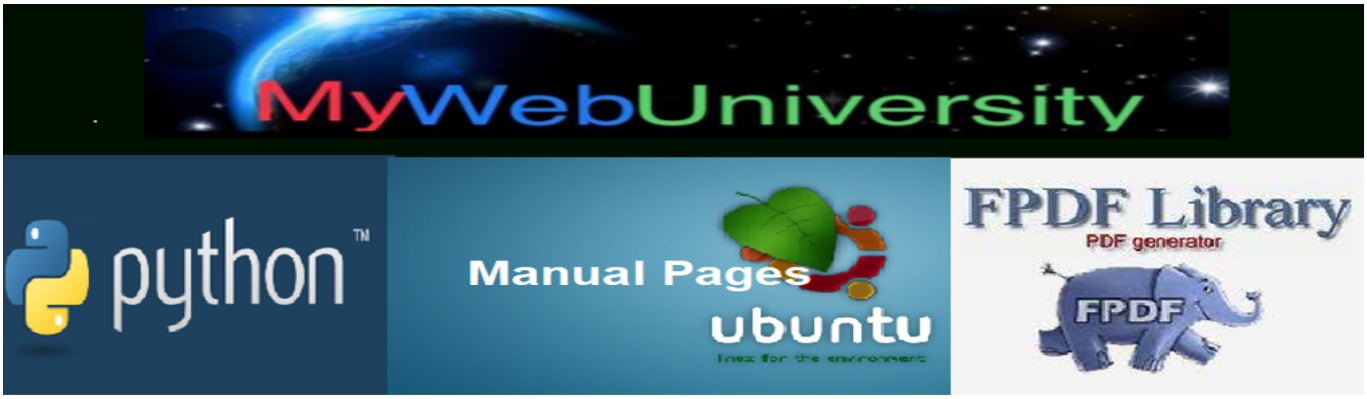

Rocky Enterprise Linux 9.2 Manual Pages on command 'openssl-format-options.1ssl'

$ man openssl-format-options.1ssl

OPENSSL-FORMAT-OPTIONS(1SSL) OpenSSL OPENSSL-FORMAT-OPTIONS(1SSL)

NAME

openssl-format-options - OpenSSL command input and output format options

SYNOPSIS

openssl command [options ...] [parameters ...]

DESCRIPTION

Several OpenSSL commands can take input or generate output in a variety of formats.

Since OpenSSL 3.0 keys, single certificates, and CRLs can be read from files in any of the DER, PEM or P12 formats. Specifying their input format is no more needed and the openssl commands will automatically try all the possible formats. However if the DER or PEM input format is specified it will be enforced.

In order to access a key via an engine the input format ENGINE may be used; alternatively the key identifier in the <uri> argument of the respective key option may be preceded by "org.openssl.engine:". See "Engine Options" in openssl(1) for an example usage of the latter.

OPTIONS

Format Options

The options to specify the format are as follows. Refer to the individual man page to see which options are accepted.

-inform format, -outform format

The format of the input or output streams.

-keyform format

Format of a private key input source.

-CRLform format

Format of a CRL input source.

Format Option Arguments

The possible format arguments are described below. Both uppercase and lowercase are accepted.

The list of acceptable format arguments, and the default, is described in each command documentation.

DER A binary format, encoded or parsed according to Distinguished Encoding Rules (DER) of the ASN.1 data language.

ENGINE

Used to specify that the cryptographic material is in an OpenSSL engine. An engine must be configured or specified using the -engine option. A password or PIN may be supplied to the engine using the -passin option.

P12 A DER-encoded file containing a PKCS#12 object. It might be necessary to provide a decryption password to retrieve the private key.

PEM A text format defined in IETF RFC 1421 and IETF RFC 7468. Briefly, this is a block of base-64 encoding (defined in IETF RFC 4648), with specific lines used to mark the start and end:

Text before the BEGIN line is ignored.

----- BEGIN object-type -----

OT43gQKBgQC/2OHZoko6iRlNOAQ/tMVFNq7fL81GivoQ9F1U0Qr+DH3ZfaH8eIkX

xT0ToMPJUzWAn8pZv0snA0um6SIgvkCuxO84OkANCVbttzXImIsL7pFzfcwV/ERK

UM6j0ZuSMFOCr/lGPAoOQU0fskidGEHi1/kW+suSr28TqsyYZpwBDQ==

----- END object-type -----

Text after the END line is also ignored

The object-type must match the type of object that is expected. For example a "BEGIN X509 CERTIFICATE" will not match if the command is trying to read a private key. The types supported include:

ANY PRIVATE KEY

CERTIFICATE

CERTIFICATE REQUEST

CMS

DH PARAMETERS

DSA PARAMETERS

DSA PUBLIC KEY

EC PARAMETERS

EC PRIVATE KEY

ECDSA PUBLIC KEY

ENCRYPTED PRIVATE KEY

PARAMETERS

PKCS #7 SIGNED DATA

PKCS7

PRIVATE KEY

PUBLIC KEY

RSA PRIVATE KEY

SSL SESSION PARAMETERS

TRUSTED CERTIFICATE

X509 CRL

X9.42 DH PARAMETERS

The following legacy object-type's are also supported for compatibility with earlier releases:

DSA PRIVATE KEY

NEW CERTIFICATE REQUEST

RSA PUBLIC KEY

X509 CERTIFICATE

SMIME

An S/MIME object as described in IETF RFC 8551. Earlier versions were known as CMS and are compatible. Note that the parsing is simple and might fail to parse some legal data.

COPYRIGHT