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

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'PKCS12_key_gen_asc.3ossl' command

$ man PKCS12_key_gen_asc.3ossl

PKCS12_KEY_GEN_UTF8_EX(3ossl) OpenSSL PKCS12_KEY_GEN_UTF8_EX(3ossl)

NAME

PKCS12_key_gen_asc, PKCS12_key_gen_asc_ex, PKCS12_key_gen_uni, PKCS12_key_gen_uni_ex, PKCS12_key_gen_utf8, PKCS12_key_gen_utf8_ex - PKCS#12 Password based key derivation

SYNOPSIS

```
#include <openssl/pkcs12.h>

int PKCS12_key_gen_asc(const char *pass, int passlen, unsigned char *salt,
                       int saltlen, int id, int iter, int n,
                       unsigned char *out, const EVP_MD *md_type);
int PKCS12_key_gen_asc_ex(const char *pass, int passlen, unsigned char *salt,
                          int saltlen, int id, int iter, int n,
                          unsigned char *out, const EVP_MD *md_type,
                          OSSL_LIB_CTX *ctx, const char *propq);
int PKCS12_key_gen_uni(unsigned char *pass, int passlen, unsigned char *salt,
                       int saltlen, int id, int iter, int n,
                       unsigned char *out, const EVP_MD *md_type);
int PKCS12_key_gen_uni_ex(unsigned char *pass, int passlen, unsigned char *salt,
                          int saltlen, int id, int iter, int n,
                          unsigned char *out, const EVP_MD *md_type,
```

```
                    OSSL_LIB_CTX *ctx, const char *propq);
int PKCS12_key_gen_utf8(const char *pass, int passlen, unsigned char *salt,
                        int saltlen, int id, int iter, int n,
                        unsigned char *out, const EVP_MD *md_type);
int PKCS12_key_gen_utf8_ex(const char *pass, int passlen, unsigned char *salt,
                           int saltlen, int id, int iter, int n,
                           unsigned char *out, const EVP_MD *md_type,
                           OSSL_LIB_CTX *ctx, const char *propq);
```

DESCRIPTION

These methods perform a key derivation according to PKCS#12 (RFC7292) with an input password pass of length passlen, a salt salt of length saltlen, an iteration count iter and a digest algorithm md_type. The ID byte id determines how the resulting key is intended to be used:

- If ID=1, then the pseudorandom bits being produced are to be used as key material for performing encryption or decryption.
- If ID=2, then the pseudorandom bits being produced are to be used as an IV (Initial Value) for encryption or decryption.
- If ID=3, then the pseudorandom bits being produced are to be used as an integrity key for MACing.

The intended format of the supplied password is determined by the method chosen:

- PKCS12_key_gen_asc() and PKCS12_key_gen_asc_ex() expect an ASCII-formatted password.
- PKCS12_key_gen_uni() and PKCS12_key_gen_uni_ex() expect a Unicode-formatted password.

? PKCS12_key_gen_utf8() and PKCS12_key_gen_utf8_ex() expect a UTF-8 encoded password.

pass is the password used in the derivation of length passlen. pass is an optional parameter and can be NULL. If passlen is -1, then the function will calculate the length of pass using strlen().

salt is the salt used in the derivation of length saltlen. If the salt is NULL, then saltlen must be 0. The function will not attempt to calculate the length of the salt because it is not assumed to be NULL terminated.

iter is the iteration count and its value should be greater than or equal to 1. RFC 2898 suggests an iteration count of at least 1000. Any iter less than 1 is treated as a single iteration.

digest is the message digest function used in the derivation.

The derived key will be written to out. The size of the out buffer is specified via n.

Functions ending in _ex() allow for a library context ctx and property query propq to be used to select algorithm implementations.

NOTES

A typical application of this function is to derive keying material for an encryption algorithm from a password in the pass, a salt in salt, and an iteration count.

Increasing the iter parameter slows down the algorithm which makes it harder for an attacker to perform a brute force attack using a large number of candidate passwords.

RETURN VALUES

Returns 1 on success or 0 on error.

CONFORMING TO

IETF RFC 7292 (<https://tools.ietf.org/html/rfc7292>)

SEE ALSO

PKCS12_create_ex(3), PKCS12_pbe_crypt_ex(3), passphrase-encoding(7)

HISTORY

PKCS12_key_gen_asc_ex(), PKCS12_key_gen_uni_ex() and
PKCS12_key_gen_utf8_ex() were added in OpenSSL 3.0.

COPYRIGHT