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

*Hat Enterprise Linux Release 9.2 Manual Pages on 'OSSL_ENCODER_CTX_set_passphrase_cb.3ossl' comm

$ man OSSL_ENCODER_CTX_set_passphrase_cb.3ossl

OSSL_ENCODER_CTX_NEW_FOR_PKEY(3ossl)OpenSSOSSL_ENCODER_CTX_NEW_FOR_PKEY(3ossl)

NAME

OSSL_ENCODER_CTX_new_for_pkey, OSSL_ENCODER_CTX_set_cipher,

OSSL_ENCODER_CTX_set_passphrase, OSSL_ENCODER_CTX_set_pem_password_cb,

OSSL_ENCODER_CTX_set_passphrase_cb, OSSL_ENCODER_CTX_set_passphrase_ui

- Encoder routines to encode EVP_PKEYs

SYNOPSIS

```
#include <openssl/encoder.h>
OSSL_ENCODER_CTX *
OSSL_ENCODER_CTX_new_for_pkey(const EVP_PKEY *pkey, int selection,
                              const char *output_type,
                              const char *output_structure,
                              const char *propquery);
int OSSL_ENCODER_CTX_set_cipher(OSSL_ENCODER_CTX *ctx,
                                const char *cipher_name,
                                const char *propquery);
int OSSL_ENCODER_CTX_set_passphrase(OSSL_ENCODER_CTX *ctx,
                                   const unsigned char *kstr,
                                   size_t klen);
int OSSL_ENCODER_CTX_set_pem_password_cb(OSSL_ENCODER_CTX *ctx,
                                        pem_password_cb *cb, void *cbarg);
int OSSL_ENCODER_CTX_set_passphrase_ui(OSSL_ENCODER_CTX *ctx,
                                       const UI_METHOD *ui_method,
```

```
                                    void *ui_data);
 int OSSL_ENCODER_CTX_set_passphrase_cb(OSSL_ENCODER_CTX *ctx,
                                    OSSL_PASSPHRASE_CALLBACK *cb,
                                    void *cbarg);
```

DESCRIPTION

OSSL_ENCODER_CTX_new_for_pkey() is a utility function that creates a OSSL_ENCODER_CTX, finds all applicable encoder implementations and sets them up, so almost all the caller has to do next is call functions like OSSL_ENCODER_to_bio(3). output_type determines the final output encoding, and selection can be used to select what parts of the pkey should be included in the output. output_type is further discussed in "Output types" below, and selection is further described in "Selections".

Internally, OSSL_ENCODER_CTX_new_for_pkey() uses the names from the EVP_KEYMGMT(3) implementation associated with pkey to build a list of applicable encoder implementations that are used to process the pkey into the encoding named by output_type, with the outermost structure named by output_structure if that's relevant. All these implementations are implicitly fetched, with propquery for finer selection.

If no suitable encoder implementation is found, OSSL_ENCODER_CTX_new_for_pkey() still creates a OSSL_ENCODER_CTX, but with no associated encoder (OSSL_ENCODER_CTX_get_num_encoders(3) returns zero). This helps the caller to distinguish between an error when creating the OSSL_ENCODER_CTX and missing encoder implementation, and allows it to act accordingly.

OSSL_ENCODER_CTX_set_cipher() tells the implementation what cipher should be used to encrypt encoded keys. The cipher is given by name cipher_name. The interpretation of that cipher_name is implementation dependent. The implementation may implement the cipher directly itself or by other implementations, or it may choose to fetch it. If the implementation supports fetching the cipher, then it may use propquery as properties to be queried for when fetching. cipher_name may also be

NULL, which will result in unencrypted encoding.

OSSL_ENCODER_CTX_set_passphrase() gives the implementation a pass phrase to use when encrypting the encoded private key. Alternatively, a pass phrase callback may be specified with the following functions.

OSSL_ENCODER_CTX_set_pem_password_cb(), OSSL_ENCODER_CTX_set_passphrase_ui() and OSSL_ENCODER_CTX_set_passphrase_cb() sets up a callback method that the implementation can use to prompt for a pass phrase, giving the caller the choice of prefered pass phrase callback form. These are called indirectly, through an internal OSSL_PASSPHRASE_CALLBACK function.

Output types

The possible EVP_PKEY output types depends on the available implementations.

OpenSSL has built in implementations for the following output types:

"TEXT"

The output is a human readable description of the key.

EVP_PKEY_print_private(3), EVP_PKEY_print_public(3) and EVP_PKEY_print_params(3) use this for their output.

"DER"

The output is the DER encoding of the selection of the pkey.

"PEM"

The output is the selection of the pkey in PEM format.

Selections

selection can be any one of the values described in "Selections" in EVP_PKEY_fromdata(3).

These are only 'hints' since the encoder implementations are free to determine what makes sense to include in the output, and this may depend on the desired output. For example, an EC key in a PKCS#8 structure doesn't usually include the public key.

RETURN VALUES

OSSL_ENCODER_CTX_new_for_pkey() returns a pointer to an OSSL_ENCODER_CTX, or NULL if it couldn't be created.

OSSL_ENCODER_CTX_set_cipher(), OSSL_ENCODER_CTX_set_passphrase(),

OSSL_ENCODER_CTX_set_pem_password_cb(),

OSSL_ENCODER_CTX_set_passphrase_ui() and

OSSL_ENCODER_CTX_set_passphrase_cb() all return 1 on success, or 0 on

failure.

SEE ALSO

provider(7), OSSL_ENCODER(3), OSSL_ENCODER_CTX(3)

HISTORY

The functions described here were added in OpenSSL 3.0.

COPYRIGHT