

NAME

`EVP_PKEY_encrypt_init_ex`, `EVP_PKEY_encrypt_init`, `EVP_PKEY_encrypt` - encrypt using a public key algorithm

SYNOPSIS

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

```
int EVP_PKEY_encrypt_init(EVP_PKEY_CTX *ctx);
int EVP_PKEY_encrypt_init_ex(EVP_PKEY_CTX *ctx, const OSSL_PARAM params[]);
int EVP_PKEY_encrypt(EVP_PKEY_CTX *ctx,
                    unsigned char *out, size_t *outlen,
                    const unsigned char *in, size_t inlen);
```

DESCRIPTION

The `EVP_PKEY_encrypt_init()` function initializes a public key algorithm context using key **pkey** for an encryption operation.

The `EVP_PKEY_encrypt_init_ex()` function initializes a public key algorithm context using key **pkey** for an encryption operation and sets the algorithm specific **params**.

The `EVP_PKEY_encrypt()` function performs a public key encryption operation using **ctx**. The data to be encrypted is specified using the **in** and **inlen** parameters. If **out** is **NULL** then the maximum size of the output buffer is written to the **outlen** parameter. If **out** is not **NULL** then before the call the **outlen** parameter should contain the length of the **out** buffer, if the call is successful the encrypted data is written to **out** and the amount of data written to **outlen**.

NOTES

After the call to `EVP_PKEY_encrypt_init()` algorithm specific control operations can be performed to set any appropriate parameters for the operation. These operations can be included in the `EVP_PKEY_encrypt_init_ex()` call.

The function `EVP_PKEY_encrypt()` can be called more than once on the same context if several operations are performed using the same parameters.

RETURN VALUES

`EVP_PKEY_encrypt_init()`, `EVP_PKEY_encrypt_init_ex()` and `EVP_PKEY_encrypt()` return 1 for success and 0 or a negative value for failure. In particular a return value of -2 indicates the operation is not supported by the public key algorithm.

EXAMPLES

Encrypt data using OAEP (for RSA keys). See also **PEM_read_PUBKEY(3)** or **d2i_X509(3)** for means to load a public key. You may also simply set 'eng = NULL;' to start with the default OpenSSL RSA implementation:

```
#include <openssl/evp.h>
#include <openssl/rsa.h>
#include <openssl/engine.h>

EVP_PKEY_CTX *ctx;
ENGINE *eng;
unsigned char *out, *in;
size_t outlen, inlen;
EVP_PKEY *key;

/*
 * NB: assumes eng, key, in, inlen are already set up,
 * and that key is an RSA public key
 */
ctx = EVP_PKEY_CTX_new(key, eng);
if (!ctx)
    /* Error occurred */
if (EVP_PKEY_encrypt_init(ctx) <= 0)
    /* Error */
if (EVP_PKEY_CTX_set_rsa_padding(ctx, RSA_PKCS1_OAEP_PADDING) <= 0)
    /* Error */

/* Determine buffer length */
if (EVP_PKEY_encrypt(ctx, NULL, &outlen, in, inlen) <= 0)
    /* Error */

out = OPENSSL_malloc(outlen);

if (!out)
    /* malloc failure */

if (EVP_PKEY_encrypt(ctx, out, &outlen, in, inlen) <= 0)
    /* Error */

/* Encrypted data is outlen bytes written to buffer out */
```

SEE ALSO

**d2i_X509(3), ENGINE_by_id(3), EVP_PKEY_CTX_new(3), EVP_PKEY_decrypt(3),
EVP_PKEY_sign(3), EVP_PKEY_verify(3), EVP_PKEY_verify_recover(3), EVP_PKEY_derive(3)**

HISTORY

These functions were added in OpenSSL 1.0.0.

COPYRIGHT

Copyright 2006-2022 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>.