

**NAME**

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(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()* 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.

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()* 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.

**EXAMPLE**

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_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\(3\)](#), [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 first added to OpenSSL 1.0.0.