



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 'EVP_PKEY_decapsulate.3oss1' command

`$ man EVP_PKEY_decapsulate.3oss1`

`EVP_PKEY_DECAPSULATE(3oss1)` `OpenSSL` `EVP_PKEY_DECAPSULATE(3oss1)`

NAME

`EVP_PKEY_decapsulate_init`, `EVP_PKEY_decapsulate` - Key decapsulation using a private key algorithm

SYNOPSIS

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

```
int EVP_PKEY_decapsulate_init(EVP_PKEY_CTX *ctx, const OSSL_PARAM params[]);
int EVP_PKEY_decapsulate(EVP_PKEY_CTX *ctx,
    unsigned char *secret, size_t *secretlen,
    const unsigned char *wrapped, size_t wrappedlen);
```

DESCRIPTION

The `EVP_PKEY_decapsulate_init()` function initializes a private key algorithm context `ctx` for a decapsulation operation and then sets the `params` on the context in the same way as calling `EVP_PKEY_CTX_set_params(3)`.

The `EVP_PKEY_decapsulate()` function performs a private key decapsulation operation using `ctx`. The data to be decapsulated is specified using the `wrapped` and `wrappedlen` parameters. If `secret` is

NULL then the maximum size of the output secret buffer is written to the *secretlen parameter. If secret is not NULL and the call is successful then the decapsulated secret data is written to secret and the amount of data written to secretlen.

NOTES

After the call to `EVP_PKEY_decapsulate_init()` algorithm specific parameters for the operation may be set or modified using `EVP_PKEY_CTX_set_params(3)`.

RETURN VALUES

`EVP_PKEY_decapsulate_init()` and `EVP_PKEY_decapsulate()` 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 private key algorithm.

EXAMPLES

Decapsulate data using RSA:

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

/*
 * NB: assumes rsa_priv_key is an RSA private key,
 * and that in, inlen are already set up to contain encapsulated data.
 */

EVP_PKEY_CTX *ctx = NULL;
size_t secretlen = 0;
unsigned char *secret = NULL;;

ctx = EVP_PKEY_CTX_new_from_pkey(libctx, rsa_priv_key, NULL);
if (ctx = NULL)
    /* Error */
```

```
if (EVP_PKEY_decapsulate_init(ctx, NULL) <= 0)
    /* Error */

/* Set the mode - only 'RSASVE' is currently supported */
if (EVP_PKEY_CTX_set_kem_op(ctx, "RSASVE") <= 0)
    /* Error */

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

secret = OPENSSL_malloc(secretlen);
if (secret == NULL)
    /* malloc failure */

/* Decapsulated secret data is secretlen bytes long */
if (EVP_PKEY_decapsulate(ctx, secret, &secretlen, in, inlen) <= 0)
    /* Error */
```

SEE ALSO

EVP_PKEY_CTX_new(3), EVP_PKEY_encapsulate(3), EVP_KEM-RSA(7),

HISTORY

These functions were added in OpenSSL 3.0.

COPYRIGHT

Copyright 2020-2021 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>.

