NAME

EVP_PKEY_verify_init, EVP_PKEY_verify_init_ex, EVP_PKEY_verify - signature verification using a public key algorithm

SYNOPSIS

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

int EVP_PKEY_verify_init(EVP_PKEY_CTX *ctx);
int EVP_PKEY_verify_init_ex(EVP_PKEY_CTX *ctx, const OSSL_PARAM params[]);
int EVP_PKEY_verify(EVP_PKEY_CTX *ctx,
                    const unsigned char *sig, size_t siglen,
                    const unsigned char *tbs, size_t tbslen);
```

DESCRIPTION

EVP_PKEY_verify_init() initializes a public key algorithm context *ctx* for signing using the algorithm given when the context was created using **EVP_PKEY_CTX_new**(3) or variants thereof. The algorithm is used to fetch a **EVP_SIGNATURE** method implicitly, see "Implicit fetch" in **provider**(7) for more information about implicit fetches.

EVP_PKEY_verify_init_ex() is the same as **EVP_PKEY_verify_init()** but additionally sets the passed parameters *params* on the context before returning.

The **EVP_PKEY_verify()** function performs a public key verification operation using *ctx*. The signature is specified using the *sig* and *siglen* parameters. The verified data (i.e. the data believed originally signed) is specified using the *tbs* and *tbslen* parameters.

NOTES

After the call to **EVP_PKEY_verify_init()** algorithm specific control operations can be performed to set any appropriate parameters for the operation.

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

RETURN VALUES

EVP_PKEY_verify_init() and **EVP_PKEY_verify()** return 1 if the verification was successful and 0 if it failed. Unlike other functions the return value 0 from **EVP_PKEY_verify()** only indicates that the signature did not verify successfully (that is tbs did not match the original data or the signature was of invalid form) it is not an indication of a more serious error.

A negative value indicates an error other that signature verification failure. In particular a return value

of -2 indicates the operation is not supported by the public key algorithm.

EXAMPLES

Verify signature using PKCS#1 and SHA256 digest:

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

EVP_PKEY_CTX *ctx;
unsigned char *md, *sig;
size_t mdlen, siglen;
EVP_PKEY *verify_key;

/*
 * NB: assumes verify_key, sig, siglen md and mdlen are already set up
 * and that verify_key is an RSA public key
 */
ctx = EVP_PKEY_CTX_new(verify_key, NULL /* no engine */);
if (!ctx)
    /* Error occurred */
if (EVP_PKEY_verify_init(ctx) <= 0)
    /* Error */
if (EVP_PKEY_CTX_set_rsa_padding(ctx, RSA_PKCS1_PADDING) <= 0)
    /* Error */
if (EVP_PKEY_CTX_set_signature_md(ctx, EVP_sha256()) <= 0)
    /* Error */

/* Perform operation */
ret = EVP_PKEY_verify(ctx, sig, siglen, md, mdlen);

/*
 * ret == 1 indicates success, 0 verify failure and < 0 for some
 * other error.
 */
```

SEE ALSO

EVP_PKEY_CTX_new(3), **EVP_PKEY_encrypt**(3), **EVP_PKEY_decrypt**(3), **EVP_PKEY_sign**(3), **EVP_PKEY_verify_recover**(3), **EVP_PKEY_derive**(3)

HISTORY

The **EVP_PKEY_verify_init()** and **EVP_PKEY_verify()** functions were added in OpenSSL 1.0.0.

The **EVP_PKEY_verify_init_ex()** function was added in OpenSSL 3.0.

COPYRIGHT