NAME

EVP_VerifyInit, EVP_VerifyUpdate, EVP_VerifyFinal - EVP signature verification functions

SYNOPSIS

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

int EVP_VerifyInit_ex(EVP_MD_CTX *ctx, const EVP_MD *type, ENGINE *impl);
int EVP_VerifyUpdate(EVP_MD_CTX *ctx, const void *d, unsigned int cnt);
int EVP_VerifyFinal(EVP_MD_CTX *ctx,unsigned char *sigbuf, unsigned int siglen,EVP_PKEY *pke

int EVP_VerifyInit(EVP_MD_CTX *ctx, const EVP_MD *type);
```

DESCRIPTION

The EVP signature verification routines are a high level interface to digital signatures.

EVP_VerifyInit_ex() sets up verification context **ctx** to use digest **type** from ENGINE **impl**. **ctx** must be initialized by calling *EVP_MD_CTX_init()* before calling this function.

EVP_VerifyUpdate() hashes **cnt** bytes of data at **d** into the verification context **ctx**. This function can be called several times on the same **ctx** to include additional data.

EVP_VerifyFinal() verifies the data in **ctx** using the public key **pkey** and against the **siglen** bytes at **sigbuf**.

EVP_VerifyInit() initializes verification context **ctx** to use the default implementation of digest **type**.

RETURN VALUES

EVP_VerifyInit_ex() and *EVP_VerifyUpdate()* return 1 for success and 0 for failure.

EVP_VerifyFinal() returns 1 for a correct signature, 0 for failure and -1 if some other error occurred.

The error codes can be obtained by *ERR_get_error(3)*.

NOTES

The **EVP** interface to digital signatures should almost always be used in preference to the low level interfaces. This is because the code then becomes transparent to the algorithm used and much more flexible.

Due to the link between message digests and public key algorithms the correct digest algorithm must be used with the correct public key type. A list of algorithms and associated public key algorithms appears in *EVP_DigestInit(3)*.

The call to *EVP_VerifyFinal()* internally finalizes a copy of the digest context. This means that calls to *EVP_VerifyUpdate()* and *EVP_VerifyFinal()* can be called later to digest and verify additional data.

Since only a copy of the digest context is ever finalized the context must be cleaned up after use by calling *EVP_MD_CTX_cleanup()* or a memory leak will occur.

BUGS

Older versions of this documentation wrongly stated that calls to *EVP_VerifyUpdate()* could not be made after calling *EVP_VerifyFinal()*.

Since the public key is passed in the call to *EVP_SignFinal()* any error relating to the private key (for example an unsuitable key and digest combination) will not be indicated until after potentially large amounts of data have been passed through *EVP_SignUpdate()*.

It is not possible to change the signing parameters using these function.

The previous two bugs are fixed in the newer EVP_VerifyDigest*() function.

SEE ALSO

evp(3), *EVP_SignInit(3)*, *EVP_DigestInit(3)*, *err(3)*, *evp(3)*, *hmac(3)*, *md2(3)*, *md5(3)*, *mdc2(3)*, *ripemd(3)*, *sha(3)*, *dgst(1)*

HISTORY

EVP_VerifyInit(), *EVP_VerifyUpdate()* and *EVP_VerifyFinal()* are available in all versions of SSLeay and OpenSSL.

EVP_VerifyInit_ex() was added in OpenSSL 0.9.7