NAME

evp – high–level cryptographic functions

SYNOPSIS

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

DESCRIPTION

The EVP library provides a high-level interface to cryptographic functions.

The **EVP_Seal***XXX* and **EVP_Open***XXX* functions provide public key encryption and decryption to implement digital ''envelopes''.

The **EVP_DigestSign***XXX* and **EVP_DigestVerify***XXX* functions implement digital signatures and Message Authentication Codes (MACs). Also see the older **EVP_Sign***XXX* and **EVP_Verify***XXX* functions.

Symmetric encryption is available with the **EVP_Encrypt***XXX* functions. The **EVP_Digest***XXX* functions provide message digests.

The **EVP_PKEY***XXX* functions provide a high level interface to asymmetric algorithms. To create a new EVP_PKEY see **EVP_PKEY_new** (3). EVP_PKEYs can be associated with a private key of a particular algorithm by using the functions described on the **EVP_PKEY_set1_RSA** (3) page, or new keys can be generated using **EVP_PKEY_keygen** (3). EVP_PKEYs can be compared using **EVP_PKEY_cmp** (3), or printed using **EVP_PKEY_print_private** (3).

The EVP_PKEY functions support the full range of asymmetric algorithm operations:

For key agreement see **EVP_PKEY_derive** (3)
For signing and verifying see **EVP_PKEY_sign** (3), **EVP_PKEY_verify** (3) and **EVP_PKEY_verify_recover** (3). However, note that these functions do not perform a digest of the data to be signed. Therefore normally you would use the **EVP_DigestSignInit** (3) functions for this purpose.
For encryption and decryption see **EVP_PKEY_encrypt** (3) and **EVP_PKEY_decrypt** (3) respectively. However, note that these functions perform encryption and decryption only. As public key encryption is an expensive operation, normally you would wrap an encrypted message in a ''digital envelope'' using the **EVP_SealInit** (3) and **EVP_OpenInit** (3) functions.

The **EVP_BytesToKey** (3) function provides some limited support for password based encryption. Careful selection of the parameters will provide a PKCS#5 PBKDF1 compatible implementation. However, new applications should not typically use this (preferring, for example, PBKDF2 from PCKS#5).

The **EVP_Encode***XXX* and **EVP_Decode***XXX* functions implement base 64 encoding and decoding.

All the symmetric algorithms (ciphers), digests and asymmetric algorithms (public key algorithms) can be replaced by ENGINE modules providing alternative implementations. If ENGINE implementations of ciphers or digests are registered as defaults, then the various EVP functions will automatically use those implementations automatically in preference to built in software implementations. For more information, consult the **engine** (3) man page.

Although low level algorithm specific functions exist for many algorithms their use is discouraged. They cannot be used with an ENGINE and ENGINE versions of new algorithms cannot be accessed using the low level functions. Also makes code harder to adapt to new algorithms and some options are not cleanly supported at the low level and some operations are more efficient using the high level interface.

SEE ALSO

EVP_DigestInit (3), **EVP_EncryptInit** (3), **EVP_OpenInit** (3), **EVP_SealInit** (3), **EVP_DigestSignInit** (3), **EVP_SignInit** (3), **EVP_VerifyInit** (3), **EVP_EncodeInit** (3), **EVP_PKEY_new** (3), **EVP_PKEY_set1_RSA** (3), **EVP_PKEY_keygen** (3), **EVP_PKEY_print_private** (3), **EVP_PKEY_decrypt** (3), **EVP_PKEY_encrypt** (3), **EVP_PKEY_sign** (3), **EVP_PKEY_verify** (3), **EVP_PKEY_verify_recover** (3), **EVP_PKEY_derive** (3), **EVP_BytesToKey** (3), **ENGINE_by_id** (3)

COPYRIGHT