NAME

EVP_BytesToKey - password based encryption routine

SYNOPSIS

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

int EVP_BytesToKey(const EVP_CIPHER *type,const EVP_MD *md,
const unsigned char *salt,
const unsigned char *data, int datal, int count,
unsigned char *key,unsigned char *iv);
```

DESCRIPTION

EVP_BytesToKey() derives a key and IV from various parameters. **type** is the cipher to derive the key and IV for. **md** is the message digest to use. The **salt** parameter is used as a salt in the derivation: it should point to an 8 byte buffer or NULL if no salt is used. **data** is a buffer containing **datal** bytes which is used to derive the keying data. **count** is the iteration count to use. The derived key and IV will be written to **key** and **iv** respectively.

NOTES

A typical application of this function is to derive keying material for an encryption algorithm from a password in the **data** parameter.

Increasing the **count** parameter slows down the algorithm which makes it harder for an attacker to peform a brute force attack using a large number of candidate passwords.

If the total key and IV length is less than the digest length and **MD5** is used then the derivation algorithm is compatible with PKCS#5 v1.5 otherwise a non standard extension is used to derive the extra data.

Newer applications should use more standard algorithms such as PBKDF2 as defined in PKCS#5v2.1 for key derivation.

KEY DERIVATION ALGORITHM

The key and IV is derived by concatenating D_1, D_2, etc until enough data is available for the key and IV. D_i is defined as:

```
D_i = HASHcount(D_(i-1) || data || salt)
```

where || denotes concatentaion, D_0 is empty, HASH is the digest algorithm in use, HASH^1(data) is simply HASH(data), HASH^2(data) is HASH(HASH(data)) and so on.

The initial bytes are used for the key and the subsequent bytes for the IV.

RETURN VALUES

If **data** is NULL, then *EVP_BytesToKey()* returns the number of bytes needed to store the derived key. Otherwise,*EVP_BytesT oKey()* returns the size of the derived key in bytes, or 0 on error.

SEE ALSO

evp(3), *rand(3)*, *EVP_EncryptInit(3)*

HISTORY