NAME

EVP_PKEY_set1_RSA, EVP_PKEY_set1_DSA, EVP_PKEY_set1_DH, EVP_PKEY_set1_EC_KEY, EVP_PKEY_get1_RSA, EVP_PKEY_get1_DSA, EVP_PKEY_get1_DH, EVP_PKEY_get1_EC_KEY, EVP_PKEY_get0_RSA, EVP_PKEY_get0_DSA, EVP_PKEY_get0_DH, EVP_PKEY_get0_EC_KEY, EVP_PKEY_assign_RSA, EVP_PKEY_assign_DSA, EVP_PKEY_assign_DH, EVP_PKEY_assign_EC_KEY, EVP_PKEY_assign_POLY1305, EVP_PKEY_assign_SIPHASH, EVP_PKEY_get0_hmac, EVP_PKEY_get0_poly1305, EVP_PKEY_get0_siphash, EVP_PKEY_get0, EVP_PKEY_type, EVP_PKEY_get_id, EVP_PKEY_get_base_id, EVP_PKEY_set1_engine, EVP_PKEY_get0_engine, EVP_PKEY_id, EVP_PKEY_base_id - EVP_PKEY assignment functions

SYNOPSIS

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

int EVP_PKEY_get_id(const EVP_PKEY *pkey);
int EVP_PKEY_get_base_id(const EVP_PKEY *pkey);
int EVP_PKEY_type(int type);

#define EVP_PKEY_id EVP_PKEY_get_id
#define EVP_PKEY_base_id EVP_PKEY_get_base_id
```

The following functions have been deprecated since OpenSSL 3.0, and can be hidden entirely by defining **OPENSSL_API_COMPAT** with a suitable version value, see **openssl_user_macros**(7):

```
int EVP_PKEY_set1_RSA(EVP_PKEY *pkey, RSA *key);
int EVP_PKEY_set1_DSA(EVP_PKEY *pkey, DSA *key);
int EVP_PKEY_set1_DH(EVP_PKEY *pkey, DH *key);
int EVP_PKEY_set1_EC_KEY(EVP_PKEY *pkey, EC_KEY *key);

RSA *EVP_PKEY_get1_RSA(EVP_PKEY *pkey);
DSA *EVP_PKEY_get1_DSA(EVP_PKEY *pkey);
DH *EVP_PKEY_get1_DH(EVP_PKEY *pkey);
EC_KEY *EVP_PKEY_get1_EC_KEY(EVP_PKEY *pkey);

const unsigned char *EVP_PKEY_get0_hmac(const EVP_PKEY *pkey, size_t *len);
const unsigned char *EVP_PKEY_get0_poly1305(const EVP_PKEY *pkey, size_t *len);
const unsigned char *EVP_PKEY_get0_siphash(const EVP_PKEY *pkey, size_t *len);
const RSA *EVP_PKEY_get0_RSA(const EVP_PKEY *pkey);
const DSA *EVP_PKEY_get0_DSA(const EVP_PKEY *pkey);
const DH *EVP_PKEY_get0_DH(const EVP_PKEY *pkey);
```

```
const EC_KEY *EVP_PKEY_get0_EC_KEY(const EVP_PKEY *pkey);
void *EVP_PKEY_get0(const EVP_PKEY *pkey);

int EVP_PKEY_assign_RSA(EVP_PKEY *pkey, RSA *key);
int EVP_PKEY_assign_DSA(EVP_PKEY *pkey, DSA *key);
int EVP_PKEY_assign_DH(EVP_PKEY *pkey, DH *key);
int EVP_PKEY_assign_EC_KEY(EVP_PKEY *pkey, EC_KEY *key);
int EVP_PKEY_assign_POLY1305(EVP_PKEY *pkey, ASN1_OCTET_STRING *key);
int EVP_PKEY_assign_SIPHASH(EVP_PKEY *pkey, ASN1_OCTET_STRING *key);

ENGINE *EVP_PKEY_get0_engine(const EVP_PKEY *pkey);
int EVP_PKEY_set1_engine(EVP_PKEY *pkey, ENGINE *engine);
```

DESCRIPTION

EVP_PKEY_get_base_id() returns the type of *pkey*. For example an RSA key will return **EVP_PKEY_RSA**.

EVP_PKEY_get_id() returns the actual NID associated with *pkey* only if the *pkey* type isn't implemented just in a **provider**(7). Historically keys using the same algorithm could use different NIDs. For example an RSA key could use the NIDs corresponding to the NIDs **NID_rsaEncryption** (equivalent to **EVP_PKEY_RSA**) or **NID_rsa** (equivalent to **EVP_PKEY_RSA2**). The use of alternative non-standard NIDs is now rare so **EVP_PKEY_RSA2** et al are not often seen in practice. **EVP_PKEY_get_id()** returns -1 (**EVP_PKEY_KEYMGMT**) if the *pkey* is only implemented in a **provider**(7).

EVP_PKEY_type() returns the underlying type of the NID *type*. For example EVP_PKEY_type(EVP_PKEY_RSA2) will return **EVP_PKEY_RSA**.

EVP_PKEY_set1_RSA(), **EVP_PKEY_set1_DSA()**, **EVP_PKEY_set1_DH()** and **EVP_PKEY_set1_EC_KEY()** set the key referenced by *pkey* to *key*. These functions are deprecated. Applications should instead use **EVP_PKEY_fromdata**(3).

EVP_PKEY_assign_RSA(), **EVP_PKEY_assign_DSA()**, **EVP_PKEY_assign_DH()**, **EVP_PKEY_assign_EC_KEY()**, **EVP_PKEY_assign_POLY1305()** and **EVP_PKEY_assign_SIPHASH()** set the referenced key to *key* however these use the supplied *key* internally and so *key* will be freed when the parent *pkey* is freed. These macros are deprecated. Applications should instead read an EVP_PKEY directly using the OSSL_DECODER APIs (see **OSSL_DECODER_CTX_new_for_pkey**(3)), or construct an EVP_PKEY from data using **EVP_PKEY_fromdata**(3).

EVP_PKEY_get1_RSA(), **EVP_PKEY_get1_DSA()**, **EVP_PKEY_get1_DH()** and **EVP_PKEY_get1_EC_KEY()** return the referenced key in *pkey* or NULL if the key is not of the correct type. The returned key must be freed after use. These functions are deprecated. Applications should instead use the EVP_PKEY directly where possible. If access to the low level key parameters is required then applications should use **EVP_PKEY_get_params**(3) and other similar functions. To write an EVP_PKEY out use the OSSL_ENCODER APIs (see **OSSL_ENCODER_CTX_new_for_pkey**(3)).

EVP_PKEY_get0_hmac(), **EVP_PKEY_get0_poly1305()**, **EVP_PKEY_get0_siphash()**, **EVP_PKEY_get0_RSA()**, **EVP_PKEY_get0_DSA()**, **EVP_PKEY_get0_DH()** and **EVP_PKEY_get0_EC_KEY()** return the referenced key in *pkey* or NULL if the key is not of the correct type. The reference count of the returned key is **not** incremented and so the key must not be freed after use. These functions are deprecated. Applications should instead use the EVP_PKEY directly where possible. If access to the low level key parameters is required then applications should use **EVP_PKEY_get_params**(3) and other similar functions. To write an EVP_PKEY out use the OSSL_ENCODER APIs (see **OSSL_ENCODER_CTX_new_for_pkey**(3)). **EVP_PKEY_get0()** returns a pointer to the legacy key or NULL if the key is not legacy.

Note that if an EVP_PKEY was not constructed using one of the deprecated functions such as **EVP_PKEY_set1_RSA()**, **EVP_PKEY_set1_DSA()**, **EVP_PKEY_set1_DH()** or **EVP_PKEY_set1_EC_KEY()**, or via the similarly named **EVP_PKEY_assign** macros described above then the internal key will be managed by a provider (see **provider**(7)). In that case the key returned by **EVP_PKEY_get1_RSA()**, **EVP_PKEY_get1_DSA()**, **EVP_PKEY_get1_DH()**, **EVP_PKEY_get1_EC_KEY()**, **EVP_PKEY_get0_hmac()**, **EVP_PKEY_get0_poly1305()**, **EVP_PKEY_get0_siphash()**, **EVP_PKEY_get0_RSA()**, **EVP_PKEY_get0_DSA()**, **EVP_PKEY_get0_DH()** or **EVP_PKEY_get0_EC_KEY()** will be a cached copy of the provider's key. Subsequent updates to the provider's key will not be reflected back in the cached copy, and updates made by an application to the returned key will not be reflected back in the provider's key. Subsequent calls to **EVP_PKEY_get1_RSA()**, **EVP_PKEY_get1_DSA()**, **EVP_PKEY_get1_DH()** and **EVP_PKEY_get1_EC_KEY()** will always return the cached copy returned by the first call.

EVP_PKEY_get0_engine() returns a reference to the ENGINE handling *pkey*. This function is deprecated. Applications should use providers instead of engines (see **provider**(7) for details).

EVP_PKEY_set1_engine() sets the ENGINE handling *pkey* to *engine*. It must be called after the key algorithm and components are set up. If *engine* does not include an **EVP_PKEY_METHOD** for *pkey* an error occurs. This function is deprecated. Applications should use providers instead of engines (see **provider**(7) for details).

WARNINGS

The following functions are only reliable with **EVP_PKEY**s that have been assigned an internal key

with EVP_PKEY_assign_*():

EVP_PKEY_get_id(), **EVP_PKEY_get_base_id()**, **EVP_PKEY_type()**

For EVP_PKEY key type checking purposes, **EVP_PKEY_is_a**(3) is more generic.

For purposes of retrieving the name of the **EVP_PKEY** the function **EVP_PKEY_get0_type_name**(3) is more generally useful.

The keys returned from the functions **EVP_PKEY_get0_RSA()**, **EVP_PKEY_get0_DSA()**, **EVP_PKEY_get0_DH()** and **EVP_PKEY_get0_EC_KEY()** were changed to have a "const" return type in OpenSSL 3.0. As described above the keys returned may be cached copies of the key held in a provider. Due to this, and unlike in earlier versions of OpenSSL, they should be considered read-only copies of the key. Updates to these keys will not be reflected back in the provider side key. The **EVP_PKEY_get1_RSA()**, **EVP_PKEY_get1_DSA()**, **EVP_PKEY_get1_DH()** and **EVP_PKEY_get1_EC_KEY()** functions were not changed to have a "const" return type in order that applications can "free" the return value. However applications should still consider them as read-only copies.

NOTES

In accordance with the OpenSSL naming convention the key obtained from or assigned to the *pkey* using the **1** functions must be freed as well as *pkey*.

EVP_PKEY_assign_RSA(), **EVP_PKEY_assign_DSA()**, **EVP_PKEY_assign_DH()**, **EVP_PKEY_assign_EC_KEY()**, **EVP_PKEY_assign_POLY1305()** and **EVP_PKEY_assign_SIPHASH()** are implemented as macros.

EVP_PKEY_assign_EC_KEY() looks at the curve name id to determine if the passed **EC_KEY** is an **SM2**(7) key, and will set the **EVP_PKEY** type to **EVP_PKEY_SM2** in that case, instead of **EVP_PKEY_EC**.

Most applications wishing to know a key type will simply call **EVP_PKEY_get_base_id()** and will not care about the actual type: which will be identical in almost all cases.

Previous versions of this document suggested using EVP_PKEY_type(pkey->type) to determine the type of a key. Since **EVP_PKEY** is now opaque this is no longer possible: the equivalent is EVP_PKEY_get_base_id(pkey).

EVP_PKEY_set1_engine() is typically used by an ENGINE returning an HSM key as part of its routine to load a private key.

RETURN VALUES

EVP_PKEY_set1_RSA(), **EVP_PKEY_set1_DSA()**, **EVP_PKEY_set1_DH()** and **EVP_PKEY_set1_EC_KEY()** return 1 for success or 0 for failure.

EVP_PKEY_get1_RSA(), **EVP_PKEY_get1_DSA()**, **EVP_PKEY_get1_DH()** and **EVP_PKEY_get1_EC_KEY()** return the referenced key or NULL if an error occurred.

EVP_PKEY_assign_RSA(), **EVP_PKEY_assign_DSA()**, **EVP_PKEY_assign_DH()**, **EVP_PKEY_assign_EC_KEY()**, **EVP_PKEY_assign_POLY1305()** and **EVP_PKEY_assign_SIPHASH()** return 1 for success and 0 for failure.

EVP_PKEY_get_base_id(), **EVP_PKEY_get_id()** and **EVP_PKEY_type()** return a key type or **NID_undef** (equivalently **EVP_PKEY_NONE**) on error.

EVP_PKEY_set1_engine() returns 1 for success and 0 for failure.

SEE ALSO

EVP_PKEY_new(3), **SM2**(7)

HISTORY

The **EVP_PKEY_id()** and **EVP_PKEY_base_id()** functions were renamed to include "get" in their names in OpenSSL 3.0, respectively. The old names are kept as non-deprecated alias macros.

EVP_PKEY_set1_RSA, EVP_PKEY_set1_DSA, EVP_PKEY_set1_DH, EVP_PKEY_set1_EC_KEY, EVP_PKEY_get1_RSA, EVP_PKEY_get1_DSA, EVP_PKEY_get1_DH, EVP_PKEY_get1_EC_KEY, EVP_PKEY_get0_RSA, EVP_PKEY_get0_DSA, EVP_PKEY_get0_DH, EVP_PKEY_get0_EC_KEY, EVP_PKEY_assign_RSA, EVP_PKEY_assign_DSA, EVP_PKEY_assign_DH, EVP_PKEY_assign_EC_KEY, EVP_PKEY_assign_POLY1305, EVP_PKEY_assign_SIPHASH, EVP_PKEY_get0_hmac, EVP_PKEY_get0_poly1305, EVP_PKEY_get0_siphash, EVP_PKEY_set1_engine and EVP_PKEY_get0_engine were deprecated in OpenSSL 3.0.

The return value from EVP_PKEY_get0_RSA, EVP_PKEY_get0_DSA, EVP_PKEY_get0_DH, EVP_PKEY_get0_EC_KEY were made const in OpenSSL 3.0.

The function **EVP_PKEY_set_alias_type()** was previously documented on this page. It was removed in OpenSSL 3.0.

COPYRIGHT