NAME

EVP_PKEY_keygen_init, EVP_PKEY_keygen, EVP_PKEY_paramgen_init, EVP_PKEY_paramgen, EVP_PKEY_CTX_set_cb, EVP_PKEY_CTX_get_cb, EVP_PKEY_CTX_get_keygen_info, EVP_PKEY_CTX_set_app_data, EVP_PKEY_CTX_get_app_data, EVP_PKEY_gen_cb, EVP_PKEY_check, EVP_PKEY_public_check, EVP_PKEY_param_check − key and parameter generation and check functions

SYNOPSIS

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

int EVP_PKEY_keygen_init(EVP_PKEY_CTX *ctx);
int EVP_PKEY_keygen(EVP_PKEY_CTX *ctx, EVP_PKEY **ppkey);
int EVP_PKEY_paramgen_init(EVP_PKEY_CTX *ctx);
int EVP_PKEY_paramgen(EVP_PKEY_CTX *ctx, EVP_PKEY **ppkey);

typedef int EVP_PKEY_gen_cb(EVP_PKEY_CTX *ctx);

void EVP_PKEY_CTX_set_cb(EVP_PKEY_CTX *ctx, EVP_PKEY_gen_cb *cb);
EVP_PKEY_gen_cb *EVP_PKEY_CTX_get_cb(EVP_PKEY_CTX *ctx);

int EVP_PKEY_CTX_get_keygen_info(EVP_PKEY_CTX *ctx, int idx);

void EVP_PKEY_CTX_set_app_data(EVP_PKEY_CTX *ctx, void *data);
void *EVP_PKEY_CTX_get_app_data(EVP_PKEY_CTX *ctx);

int EVP_PKEY_check(EVP_PKEY_CTX *ctx);
int EVP_PKEY_public_check(EVP_PKEY_CTX *ctx);
int EVP_PKEY_param_check(EVP_PKEY_CTX *ctx);
```

DESCRIPTION

The **EVP_PKEY_keygen_init()** function initializes a public key algorithm context using key **pkey** for a key generation operation.

The **EVP_PKEY_keygen()** function performs a key generation operation, the generated key is written to **ppkey**.

The functions **EVP_PKEY_paramgen_init()** and **EVP_PKEY_paramgen()** are similar except parameters are generated.

The function **EVP_PKEY_set_cb()** sets the key or parameter generation callback to **cb**. The function **EVP_PKEY_CTX_get_cb()** returns the key or parameter generation callback.

The function **EVP_PKEY_CTX_get_keygen_info()** returns parameters associated with the generation operation. If **idx** is −1 the total number of parameters available is returned. Any non negative value returns the value of that parameter. **EVP_PKEY_CTX_gen_keygen_info()** with a nonnegative value for **idx** should only be called within the generation callback.

If the callback returns 0 then the key generation operation is aborted and an error occurs. This might occur during a time consuming operation where a user clicks on a ‘‘cancel’’ button.

The functions **EVP_PKEY_CTX_set_app_data()** and **EVP_PKEY_CTX_get_app_data()** set and retrieve an opaque pointer. This can be used to set some application defined value which can be retrieved in the callback: for example a handle which is used to update a ‘‘progress dialog’’.

EVP_PKEY_check() validates the key-pair given by **ctx**. This function first tries to use customized key check method in **EVP_PKEY_METHOD** if it’s present; otherwise it calls a default one defined in **EVP_PKEY_ASN1_METHOD**.

EVP_PKEY_public_check() validates the public component of the key-pair given by **ctx**. This function first tries to use customized key check method in **EVP_PKEY_METHOD** if it’s present; otherwise it calls a

default one defined in **EVP_PKEY_ASN1_METHOD**.

EVP_PKEY_param_check() validates the algorithm parameters of the key-pair given by **ctx**. This function first tries to use customized key check method in **EVP_PKEY_METHOD** if it's present; otherwise it calls a default one defined in **EVP_PKEY_ASN1_METHOD**.

NOTES

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

The functions **EVP_PKEY_keygen()** and **EVP_PKEY_paramgen()** can be called more than once on the same context if several operations are performed using the same parameters.

The meaning of the parameters passed to the callback will depend on the algorithm and the specific implementation of the algorithm. Some might not give any useful information at all during key or parameter generation. Others might not even call the callback.

The operation performed by key or parameter generation depends on the algorithm used. In some cases (e.g. EC with a supplied named curve) the ''generation'' option merely sets the appropriate fields in an EVP_PKEY structure.

In OpenSSL an EVP_PKEY structure containing a private key also contains the public key components and parameters (if any). An OpenSSL private key is equivalent to what some libraries call a ''key pair''. A private key can be used in functions which require the use of a public key or parameters.

RETURN VALUES

EVP_PKEY_keygen_init(), **EVP_PKEY_paramgen_init()**, **EVP_PKEY_keygen()** and **EVP_PKEY_paramgen()** return 1 for success and 0 or a negative value for failure. In particular a return value of −2 indicates the operation is not supported by the public key algorithm.

EVP_PKEY_check(), **EVP_PKEY_public_check()** and **EVP_PKEY_param_check()** return 1 for success or others for failure. They return −2 if the operation is not supported for the specific algorithm.

EXAMPLES

Generate a 2048 bit RSA key:

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

EVP_PKEY_CTX *ctx;
EVP_PKEY *pkey = NULL;

ctx = EVP_PKEY_CTX_new_id(EVP_PKEY_RSA, NULL);
if (!ctx)
    /* Error occurred */
if (EVP_PKEY_keygen_init(ctx) <= 0)
    /* Error */
if (EVP_PKEY_CTX_set_rsa_keygen_bits(ctx, 2048) <= 0)
    /* Error */

/* Generate key */
if (EVP_PKEY_keygen(ctx, &pkey) <= 0)
    /* Error */
```

Generate a key from a set of parameters:

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

EVP_PKEY_CTX *ctx;
ENGINE *eng;
EVP_PKEY *pkey = NULL, *param;
```

```
/* Assumed param, eng are set up already */
ctx = EVP_PKEY_CTX_new(param, eng);
if (!ctx)
    /* Error occurred */
if (EVP_PKEY_keygen_init(ctx) <= 0)
    /* Error */

/* Generate key */
if (EVP_PKEY_keygen(ctx, &pkey) <= 0)
    /* Error */
```

Example of generation callback for OpenSSL public key implementations:

```
/* Application data is a BIO to output status to */

EVP_PKEY_CTX_set_app_data(ctx, status_bio);

static int genpkey_cb(EVP_PKEY_CTX *ctx)
{
    char c = '*';
    BIO *b = EVP_PKEY_CTX_get_app_data(ctx);
    int p = EVP_PKEY_CTX_get_keygen_info(ctx, 0);

    if (p == 0)
        c = '.';
    if (p == 1)
        c = '+';
    if (p == 2)
        c = '*';
    if (p == 3)
        c = '\n';
    BIO_write(b, &c, 1);
    (void)BIO_flush(b);
    return 1;
}
```

SEE ALSO

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

HISTORY

These functions were added in OpenSSL 1.0.0.

EVP_PKEY_check(), **EVP_PKEY_public_check**() and **EVP_PKEY_param_check**() were added in OpenSSL 1.1.1.

COPYRIGHT