NAME

EVP_KEYMGMT, EVP_KEYMGMT_fetch, EVP_KEYMGMT_up_ref, EVP_KEYMGMT_free, EVP_KEYMGMT_get0_provider, EVP_KEYMGMT_is_a, EVP_KEYMGMT_get0_description, EVP_KEYMGMT_get0_name, EVP_KEYMGMT_do_all_provided, EVP_KEYMGMT_names_do_all, EVP_KEYMGMT_gettable_params, EVP_KEYMGMT_settable_params, EVP_KEYMGMT_gen_settable_params - EVP key management routines

SYNOPSIS

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

typedef struct evp_keymgmt_st EVP_KEYMGMT;

EVP_KEYMGMT *EVP_KEYMGMT_fetch(OSSL_LIB_CTX *ctx, const char *algorithm,
                               const char *properties);
int EVP_KEYMGMT_up_ref(EVP_KEYMGMT *keymgmt);
void EVP_KEYMGMT_free(EVP_KEYMGMT *keymgmt);
const OSSL_PROVIDER *EVP_KEYMGMT_get0_provider(const EVP_KEYMGMT *keymgmt);
int EVP_KEYMGMT_is_a(const EVP_KEYMGMT *keymgmt, const char *name);
const char *EVP_KEYMGMT_get0_name(const EVP_KEYMGMT *keymgmt);
const char *EVP_KEYMGMT_get0_description(const EVP_KEYMGMT *keymgmt);

void EVP_KEYMGMT_do_all_provided(OSSL_LIB_CTX *libctx,
                                 void (*fn)(EVP_KEYMGMT *keymgmt, void *arg),
                                 void *arg);
int EVP_KEYMGMT_names_do_all(const EVP_KEYMGMT *keymgmt,
                             void (*fn)(const char *name, void *data),
                             void *data);
const OSSL_PARAM *EVP_KEYMGMT_gettable_params(const EVP_KEYMGMT *keymgmt);
const OSSL_PARAM *EVP_KEYMGMT_settable_params(const EVP_KEYMGMT *keymgmt);
const OSSL_PARAM *EVP_KEYMGMT_gen_settable_params(const EVP_KEYMGMT *keymgmt);
```

DESCRIPTION

EVP_KEYMGMT is a method object that represents key management implementations for different cryptographic algorithms. This method object provides functionality to have providers import key material from the outside, as well as export key material to the outside. Most of the functionality can only be used internally and has no public interface, this object is simply passed into other functions when needed.

EVP_KEYMGMT_fetch() looks for an algorithm within the provider that has been loaded into the

OSSL_LIB_CTX given by *ctx*, having the name given by *algorithm* and the properties given by *properties*.

EVP_KEYMGMT_up_ref() increments the reference count for the given **EVP_KEYMGMT** *keymgmt*.

EVP_KEYMGMT_free() decrements the reference count for the given **EVP_KEYMGMT** *keymgmt*, and when the count reaches zero, frees it.

EVP_KEYMGMT_get0_provider() returns the provider that has this particular implementation.

EVP_KEYMGMT_is_a() checks if *keymgmt* is an implementation of an algorithm that's identifiable with *name*.

EVP_KEYMGMT_get0_name() returns the algorithm name from the provided implementation for the given *keymgmt*. Note that the *keymgmt* may have multiple synonyms associated with it. In this case the first name from the algorithm definition is returned. Ownership of the returned string is retained by the *keymgmt* object and should not be freed by the caller.

EVP_KEYMGMT_names_do_all() traverses all names for the *keymgmt*, and calls *fn* with each name and *data*.

EVP_KEYMGMT_get0_description() returns a description of the *keymgmt*, meant for display and human consumption. The description is at the discretion of the *keymgmt* implementation.

EVP_KEYMGMT_do_all_provided() traverses all key keymgmt implementations by all activated providers in the library context *libctx*, and for each of the implementations, calls *fn* with the implementation method and *data* as arguments.

EVP_KEYMGMT_gettable_params() and **EVP_KEYMGMT_settable_params()** return a constant **OSSL_PARAM**(3) array that describes the names and types of key parameters that can be retrieved or set. **EVP_KEYMGMT_gettable_params()** is used by **EVP_PKEY_gettable_params**(3).

EVP_KEYMGMT_gen_settable_params() returns a constant **OSSL_PARAM**(3) array that describes the names and types of key generation parameters that can be set via **EVP_PKEY_CTX_set_params**(3).

NOTES

EVP_KEYMGMT_fetch() may be called implicitly by other fetching functions, using the same library context and properties. Any other API that uses keys will typically do this.

RETURN VALUES

EVP_KEYMGMT_fetch() returns a pointer to the key management implementation represented by an EVP_KEYMGMT object, or NULL on error.

EVP_KEYMGMT_up_ref() returns 1 on success, or 0 on error.

EVP_KEYMGMT_names_do_all() returns 1 if the callback was called for all names. A return value of 0 means that the callback was not called for any names.

EVP_KEYMGMT_free() doesn't return any value.

EVP_KEYMGMT_get0_provider() returns a pointer to a provider object, or NULL on error.

EVP_KEYMGMT_is_a() returns 1 of *keymgmt* was identifiable, otherwise 0.

EVP_KEYMGMT_get0_name() returns the algorithm name, or NULL on error.

EVP_KEYMGMT_get0_description() returns a pointer to a description, or NULL if there isn't one.

EVP_KEYMGMT_gettable_params(), **EVP_KEYMGMT_settable_params()** and **EVP_KEYMGMT_gen_settable_params()** return a constant **OSSL_PARAM**(3) array or NULL on error.

SEE ALSO

EVP_MD_fetch(3), **OSSL_LIB_CTX**(3)

HISTORY

The functions described here were added in OpenSSL 3.0.

COPYRIGHT