

NAME

EVP_ASYM_CIPHER_fetch, EVP_ASYM_CIPHER_free, EVP_ASYM_CIPHER_up_ref,
 EVP_ASYM_CIPHER_is_a, EVP_ASYM_CIPHER_get0_provider,
 EVP_ASYM_CIPHER_do_all_provided, EVP_ASYM_CIPHER_names_do_all,
 EVP_ASYM_CIPHER_get0_name, EVP_ASYM_CIPHER_get0_description,
 EVP_ASYM_CIPHER_gettable_ctx_params, EVP_ASYM_CIPHER_settable_ctx_params - Functions
 to manage EVP_ASYM_CIPHER algorithm objects

SYNOPSIS

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

```
EVP_ASYM_CIPHER *EVP_ASYM_CIPHER_fetch(OSSL_LIB_CTX *ctx, const char *algorithm,
    const char *properties);
void EVP_ASYM_CIPHER_free(EVP_ASYM_CIPHER *cipher);
int EVP_ASYM_CIPHER_up_ref(EVP_ASYM_CIPHER *cipher);
const char *EVP_ASYM_CIPHER_get0_name(const EVP_ASYM_CIPHER *cipher);
int EVP_ASYM_CIPHER_is_a(const EVP_ASYM_CIPHER *cipher, const char *name);
OSSL_PROVIDER *EVP_ASYM_CIPHER_get0_provider(const EVP_ASYM_CIPHER *cipher);
void EVP_ASYM_CIPHER_do_all_provided(OSSL_LIB_CTX *libctx,
    void (*fn)(EVP_ASYM_CIPHER *cipher,
        void *arg),
    void *arg);
int EVP_ASYM_CIPHER_names_do_all(const EVP_ASYM_CIPHER *cipher,
    void (*fn)(const char *name, void *data),
    void *data);
const char *EVP_ASYM_CIPHER_get0_description(const EVP_ASYM_CIPHER *cipher);
const OSSL_PARAM *EVP_ASYM_CIPHER_gettable_ctx_params(const EVP_ASYM_CIPHER *cip);
const OSSL_PARAM *EVP_ASYM_CIPHER_settable_ctx_params(const EVP_ASYM_CIPHER *cip);
```

DESCRIPTION

EVP_ASYM_CIPHER_fetch() fetches the implementation for the given **algorithm** from any provider offering it, within the criteria given by the **properties** and in the scope of the given library context **ctx** (see **OSSL_LIB_CTX(3)**). The algorithm will be one offering functions for performing asymmetric cipher related tasks such as asymmetric encryption and decryption. See "ALGORITHM FETCHING" in **crypto(7)** for further information.

The returned value must eventually be freed with **EVP_ASYM_CIPHER_free()**.

EVP_ASYM_CIPHER_free() decrements the reference count for the **EVP_ASYM_CIPHER** structure. Typically this structure will have been obtained from an earlier call to **EVP_ASYM_CIPHER_fetch()**.

If the reference count drops to 0 then the structure is freed.

EVP_ASYM_CIPHER_up_ref() increments the reference count for an **EVP_ASYM_CIPHER** structure.

EVP_ASYM_CIPHER_is_a() returns 1 if *cipher* is an implementation of an algorithm that's identifiable with *name*, otherwise 0.

EVP_ASYM_CIPHER_get0_provider() returns the provider that *cipher* was fetched from.

EVP_ASYM_CIPHER_do_all_provided() traverses all **EVP_ASYM_CIPHER**s implemented by all activated providers in the given library context *libctx*, and for each of the implementations, calls the given function *fn* with the implementation method and the given *arg* as argument.

EVP_ASYM_CIPHER_get0_name() returns the algorithm name from the provided implementation for the given *cipher*. Note that the *cipher* 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 *cipher* object and should not be freed by the caller.

EVP_ASYM_CIPHER_names_do_all() traverses all names for *cipher*, and calls *fn* with each name and *data*.

EVP_ASYM_CIPHER_get0_description() returns a description of the *cipher*, meant for display and human consumption. The description is at the discretion of the *cipher* implementation.

EVP_ASYM_CIPHER_gettable_ctx_params() and **EVP_ASYM_CIPHER_settable_ctx_params()** return a constant **OSSL_PARAM(3)** array that describes the names and types of key parameters that can be retrieved or set by a key encryption algorithm using **EVP_PKEY_CTX_get_params(3)** and **EVP_PKEY_CTX_set_params(3)**.

RETURN VALUES

EVP_ASYM_CIPHER_fetch() returns a pointer to an **EVP_ASYM_CIPHER** for success or **NULL** for failure.

EVP_ASYM_CIPHER_up_ref() returns 1 for success or 0 otherwise.

EVP_ASYM_CIPHER_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_ASYM_CIPHER_gettable_ctx_params() and **EVP_ASYM_CIPHER_settable_ctx_params()**

return a constant **OSSL_PARAM**(3) array or NULL on error.

SEE ALSO

"ALGORITHM FETCHING" in **crypto**(7), **OSSL_PROVIDER**(3)

HISTORY

The functions described here were added in OpenSSL 3.0.

COPYRIGHT

Copyright 2019-2021 The OpenSSL Project Authors. All Rights Reserved.

Licensed under the Apache License 2.0 (the "License"). You may not use this file except in compliance with the License. You can obtain a copy in the file LICENSE in the source distribution or at <<https://www.openssl.org/source/license.html>>.