

**NAME**

EVP\_KEYEXCH\_fetch, EVP\_KEYEXCH\_free, EVP\_KEYEXCH\_up\_ref,  
 EVP\_KEYEXCH\_get0\_provider, EVP\_KEYEXCH\_is\_a, EVP\_KEYEXCH\_do\_all\_provided,  
 EVP\_KEYEXCH\_names\_do\_all, EVP\_KEYEXCH\_get0\_name, EVP\_KEYEXCH\_get0\_description,  
 EVP\_KEYEXCH\_gettable\_ctx\_params, EVP\_KEYEXCH\_settable\_ctx\_params - Functions to manage  
 EVP\_KEYEXCH algorithm objects

**SYNOPSIS**

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

```
EVP_KEYEXCH *EVP_KEYEXCH_fetch(OSSL_LIB_CTX *ctx, const char *algorithm,
                                const char *properties);
void EVP_KEYEXCH_free(EVP_KEYEXCH *exchange);
int EVP_KEYEXCH_up_ref(EVP_KEYEXCH *exchange);
OSSL_PROVIDER *EVP_KEYEXCH_get0_provider(const EVP_KEYEXCH *exchange);
int EVP_KEYEXCH_is_a(const EVP_KEYEXCH *exchange, const char *name);
const char *EVP_KEYEXCH_get0_name(const EVP_KEYEXCH *exchange);
void EVP_KEYEXCH_do_all_provided(OSSL_LIB_CTX *libctx,
                                void (*fn)(EVP_KEYEXCH *exchange, void *arg),
                                void *arg);
int EVP_KEYEXCH_names_do_all(const EVP_KEYEXCH *exchange,
                             void (*fn)(const char *name, void *data),
                             void *data);
const char *EVP_KEYEXCH_get0_description(const EVP_KEYEXCH *keyexch);
const OSSL_PARAM *EVP_KEYEXCH_gettable_ctx_params(const EVP_KEYEXCH *keyexch);
const OSSL_PARAM *EVP_KEYEXCH_settable_ctx_params(const EVP_KEYEXCH *keyexch);
```

**DESCRIPTION**

**EVP\_KEYEXCH\_fetch()** fetches the key exchange implementation for the given *algorithm* from any provider offering it, within the criteria given by the *properties*. See "ALGORITHM FETCHING" in **crypto(7)** for further information.

The returned value must eventually be freed with **EVP\_KEYEXCH\_free()**.

**EVP\_KEYEXCH\_free()** decrements the reference count for the **EVP\_KEYEXCH** structure. Typically this structure will have been obtained from an earlier call to **EVP\_KEYEXCH\_fetch()**. If the reference count drops to 0 then the structure is freed.

**EVP\_KEYEXCH\_up\_ref()** increments the reference count for an **EVP\_KEYEXCH** structure.

**EVP\_KEYEXCH\_get0\_provider()** returns the provider that *exchange* was fetched from.

**EVP\_KEYEXCH\_is\_a()** checks if *exchange* is an implementation of an algorithm that's identifiable with *name*.

**EVP\_KEYEXCH\_get0\_name()** returns the algorithm name from the provided implementation for the given *exchange*. Note that the *exchange* 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 *exchange* object and should not be freed by the caller.

**EVP\_KEYEXCH\_names\_do\_all()** traverses all names for the *exchange*, and calls *fn* with each name and *data*.

**EVP\_KEYEXCH\_get0\_description()** returns a description of the *keyexch*, meant for display and human consumption. The description is at the discretion of the *keyexch* implementation.

**EVP\_KEYEXCH\_do\_all\_provided()** traverses all key exchange 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\_KEYEXCH\_gettable\_ctx\_params()** and **EVP\_KEYEXCH\_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 exchange algorithm using **EVP\_PKEY\_CTX\_get\_params(3)** and **EVP\_PKEY\_CTX\_set\_params(3)**.

## RETURN VALUES

**EVP\_KEYEXCH\_fetch()** returns a pointer to a **EVP\_KEYEXCH** for success or NULL for failure.

**EVP\_KEYEXCH\_up\_ref()** returns 1 for success or 0 otherwise.

**EVP\_KEYEXCH\_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\_KEYEXCH\_is\_a()** returns 1 if *exchange* was identifiable, otherwise 0.

**EVP\_KEYEXCH\_gettable\_ctx\_params()** and **EVP\_KEYEXCH\_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>.