

**NAME**

provider-kdf – The KDF library <--> provider functions

**SYNOPSIS**

```
#include <openssl/core_dispatch.h>
#include <openssl/core_names.h>

/*
 * None of these are actual functions, but are displayed like this for
 * the function signatures for functions that are offered as function
 * pointers in OSSL_DISPATCH arrays.
 */

/* Context management */
void *OSSL_FUNC_kdf_newctx(void *provctx);
void OSSL_FUNC_kdf_freectx(void *kctx);
void *OSSL_FUNC_kdf_dupctx(void *src);

/* Encryption/decryption */
int OSSL_FUNC_kdf_reset(void *kctx);
int OSSL_FUNC_kdf_derive(void *kctx, unsigned char *key, size_t keylen,
                        const OSSL_PARAM params[]);

/* KDF parameter descriptors */
const OSSL_PARAM *OSSL_FUNC_kdf_gettable_params(void *provctx);
const OSSL_PARAM *OSSL_FUNC_kdf_gettable_ctx_params(void *kctx, void *provctx);
const OSSL_PARAM *OSSL_FUNC_kdf_settable_ctx_params(void *kctx, void *provctx);

/* KDF parameters */
int OSSL_FUNC_kdf_get_params(OSSL_PARAM params[]);
int OSSL_FUNC_kdf_get_ctx_params(void *kctx, OSSL_PARAM params[]);
int OSSL_FUNC_kdf_set_ctx_params(void *kctx, const OSSL_PARAM params[]);
```

**DESCRIPTION**

This documentation is primarily aimed at provider authors. See **provider**(7) for further information.

The KDF operation enables providers to implement KDF algorithms and make them available to applications via the API functions **EVP\_KDF\_CTX\_reset**(3), and **EVP\_KDF\_derive**(3).

All “functions” mentioned here are passed as function pointers between *libcrypto* and the provider in **OSSL\_DISPATCH** arrays via **OSSL\_ALGORITHM** arrays that are returned by the provider’s **provider\_query\_operation**() function (see “Provider Functions” in **provider-base**(7)).

All these “functions” have a corresponding function type definition named **OSSL\_FUNC\_{name}\_fn**, and a helper function to retrieve the function pointer from an **OSSL\_DISPATCH** element named **OSSL\_FUNC\_{name}**. For example, the “function” **OSSL\_FUNC\_kdf\_newctx**() has these:

```
typedef void *(OSSL_FUNC_kdf_newctx_fn)(void *provctx);
static openssl_inline OSSL_FUNC_kdf_newctx_fn
    OSSL_FUNC_kdf_newctx(const OSSL_DISPATCH *opf);
```

**OSSL\_DISPATCH** array entries are identified by numbers that are provided as macros in **openssl-core\_dispatch.h**(7), as follows:

<b>OSSL_FUNC_kdf_newctx</b>	<b>OSSL_FUNC_KDF_NEWCTX</b>
<b>OSSL_FUNC_kdf_freectx</b>	<b>OSSL_FUNC_KDF_FREECTX</b>
<b>OSSL_FUNC_kdf_dupctx</b>	<b>OSSL_FUNC_KDF_DUPCTX</b>
<b>OSSL_FUNC_kdf_reset</b>	<b>OSSL_FUNC_KDF_RESET</b>

<code>OSSL_FUNC_kdf_derive</code>	<code>OSSL_FUNC_KDF_DERIVE</code>
<code>OSSL_FUNC_kdf_get_params</code>	<code>OSSL_FUNC_KDF_GET_PARAMS</code>
<code>OSSL_FUNC_kdf_get_ctx_params</code>	<code>OSSL_FUNC_KDF_GET_CTX_PARAMS</code>
<code>OSSL_FUNC_kdf_set_ctx_params</code>	<code>OSSL_FUNC_KDF_SET_CTX_PARAMS</code>
<code>OSSL_FUNC_kdf_gettable_params</code>	<code>OSSL_FUNC_KDF_GETTABLE_PARAMS</code>
<code>OSSL_FUNC_kdf_gettable_ctx_params</code>	<code>OSSL_FUNC_KDF_GETTABLE_CTX_PARAMS</code>
<code>OSSL_FUNC_kdf_settable_ctx_params</code>	<code>OSSL_FUNC_KDF_SETTABLE_CTX_PARAMS</code>

A KDF algorithm implementation may not implement all of these functions. In order to be a consistent set of functions, at least the following functions must be implemented: **OSSL\_FUNC\_kdf\_newctx()**, **OSSL\_FUNC\_kdf\_freectx()**, **OSSL\_FUNC\_kdf\_set\_ctx\_params()**, **OSSL\_FUNC\_kdf\_derive()**. All other functions are optional.

### Context Management Functions

**OSSL\_FUNC\_kdf\_newctx()** should create and return a pointer to a provider side structure for holding context information during a KDF operation. A pointer to this context will be passed back in a number of the other KDF operation function calls. The parameter *provctx* is the provider context generated during provider initialisation (see **provider** (7)).

**OSSL\_FUNC\_kdf\_freectx()** is passed a pointer to the provider side KDF context in the *kctx* parameter. If it receives NULL as *kctx* value, it should not do anything other than return. This function should free any resources associated with that context.

**OSSL\_FUNC\_kdf\_dupctx()** should duplicate the provider side KDF context in the *kctx* parameter and return the duplicate copy.

### Encryption/Decryption Functions

**OSSL\_FUNC\_kdf\_reset()** initialises a KDF operation given a provider side KDF context in the *kctx* parameter.

**OSSL\_FUNC\_kdf\_derive()** performs the KDF operation after processing the *params* as per **OSSL\_FUNC\_kdf\_set\_ctx\_params()**. The *kctx* parameter contains a pointer to the provider side context. The resulting key of the desired *keylen* should be written to *key*. If the algorithm does not support the requested *keylen* the function must return error.

### KDF Parameters

See **OSSL\_PARAM** (3) for further details on the parameters structure used by these functions.

**OSSL\_FUNC\_kdf\_get\_params()** gets details of parameter values associated with the provider algorithm and stores them in *params*.

**OSSL\_FUNC\_kdf\_set\_ctx\_params()** sets KDF parameters associated with the given provider side KDF context *kctx* to *params*. Any parameter settings are additional to any that were previously set. Passing NULL for *params* should return true.

**OSSL\_FUNC\_kdf\_get\_ctx\_params()** retrieves gettable parameter values associated with the given provider side KDF context *kctx* and stores them in *params*. Passing NULL for *params* should return true.

**OSSL\_FUNC\_kdf\_gettable\_params()**, **OSSL\_FUNC\_kdf\_gettable\_ctx\_params()**, and **OSSL\_FUNC\_kdf\_settable\_ctx\_params()** all return constant **OSSL\_PARAM** arrays as descriptors of the parameters that **OSSL\_FUNC\_kdf\_get\_params()**, **OSSL\_FUNC\_kdf\_get\_ctx\_params()**, and **OSSL\_FUNC\_kdf\_set\_ctx\_params()** can handle, respectively. **OSSL\_FUNC\_kdf\_gettable\_ctx\_params()** and **OSSL\_FUNC\_kdf\_settable\_ctx\_params()** will return the parameters associated with the provider side context *kctx* in its current state if it is not NULL. Otherwise, they return the parameters associated with the provider side algorithm *provctx*.

Parameters currently recognised by built-in KDFs are as follows. Not all parameters are relevant to, or are understood by all KDFs:

- “size” (**OSSL\_KDF\_PARAM\_SIZE**) <unsigned integer>  
 Gets the output size from the associated KDF ctx. If the algorithm produces a variable amount of output, SIZE\_MAX should be returned. If the input parameters required to calculate the fixed output size have not yet been supplied, 0 should be returned indicating an error.
- “key” (**OSSL\_KDF\_PARAM\_KEY**) <octet string>  
 Sets the key in the associated KDF ctx.
- “secret” (**OSSL\_KDF\_PARAM\_SECRET**) <octet string>  
 Sets the secret in the associated KDF ctx.
- “pass” (**OSSL\_KDF\_PARAM\_PASSWORD**) <octet string>  
 Sets the password in the associated KDF ctx.
- “cipher” (**OSSL\_KDF\_PARAM\_CIPHER**) <UTF8 string>  
 “digest” (**OSSL\_KDF\_PARAM\_DIGEST**) <UTF8 string>  
 “mac” (**OSSL\_KDF\_PARAM\_MAC**) <UTF8 string>  
 Sets the name of the underlying cipher, digest or MAC to be used. It must name a suitable algorithm for the KDF that’s being used.
- “maclen” (**OSSL\_KDF\_PARAM\_MAC\_SIZE**) <octet string>  
 Sets the length of the MAC in the associated KDF ctx.
- “properties” (**OSSL\_KDF\_PARAM\_PROPERTIES**) <UTF8 string>  
 Sets the properties to be queried when trying to fetch the underlying algorithm. This must be given together with the algorithm naming parameter to be considered valid.
- “iter” (**OSSL\_KDF\_PARAM\_ITER**) <unsigned integer>  
 Sets the number of iterations in the associated KDF ctx.
- “mode” (**OSSL\_KDF\_PARAM\_MODE**) <UTF8 string>  
 Sets the mode in the associated KDF ctx.
- “pkcs5” (**OSSL\_KDF\_PARAM\_PKCS5**) <integer>  
 Enables or disables the SP800–132 compliance checks. A mode of 0 enables the compliance checks.  
 The checks performed are:  
 – the iteration count is at least 1000.  
 – the salt length is at least 128 bits.  
 – the derived key length is at least 112 bits.
- “ukm” (**OSSL\_KDF\_PARAM\_UKM**) <octet string>  
 Sets an optional random string that is provided by the sender called “partyAInfo”. In CMS this is the user keying material.
- “cekalg” (**OSSL\_KDF\_PARAM\_CEK\_ALG**) <UTF8 string>  
 Sets the CEK wrapping algorithm name in the associated KDF ctx.
- “n” (**OSSL\_KDF\_PARAM\_SCRIPT\_N**) <unsigned integer>  
 Sets the script work factor parameter N in the associated KDF ctx.
- “r” (**OSSL\_KDF\_PARAM\_SCRIPT\_R**) <unsigned integer>  
 Sets the script work factor parameter r in the associated KDF ctx.
- “p” (**OSSL\_KDF\_PARAM\_SCRIPT\_P**) <unsigned integer>  
 Sets the script work factor parameter p in the associated KDF ctx.
- “maxmem\_bytes” (**OSSL\_KDF\_PARAM\_SCRIPT\_MAXMEM**) <unsigned integer>  
 Sets the script work factor parameter maxmem in the associated KDF ctx.
- “prefix” (**OSSL\_KDF\_PARAM\_PREFIX**) <octet string>  
 Sets the prefix string using by the TLS 1.3 version of HKDF in the associated KDF ctx.

“label” (**OSSL\_KDF\_PARAM\_LABEL**) <octet string>  
Sets the label string using by the TLS 1.3 version of HKDF in the associated KDF ctx.

“data” (**OSSL\_KDF\_PARAM\_DATA**) <octet string>  
Sets the context string using by the TLS 1.3 version of HKDF in the associated KDF ctx.

“info” (**OSSL\_KDF\_PARAM\_INFO**) <octet string>  
Sets the optional shared info in the associated KDF ctx.

“seed” (**OSSL\_KDF\_PARAM\_SEED**) <octet string>  
Sets the IV in the associated KDF ctx.

“xcghash” (**OSSL\_KDF\_PARAM\_SSHKDF\_XCGHASH**) <octet string>  
Sets the xcghash in the associated KDF ctx.

“session\_id” (**OSSL\_KDF\_PARAM\_SSHKDF\_SESSION\_ID**) <octet string>  
Sets the session ID in the associated KDF ctx.

“type” (**OSSL\_KDF\_PARAM\_SSHKDF\_TYPE**) <UTF8 string>  
Sets the SSH KDF type parameter in the associated KDF ctx. There are six supported types:

**EVP\_KDF\_SSHKDF\_TYPE\_INITIAL\_IV\_CLI\_TO\_SRV**  
The Initial IV from client to server. A single char of value 65 (ASCII char 'A').

**EVP\_KDF\_SSHKDF\_TYPE\_INITIAL\_IV\_SRV\_TO\_CLI**  
The Initial IV from server to client A single char of value 66 (ASCII char 'B').

**EVP\_KDF\_SSHKDF\_TYPE\_ENCRYPTION\_KEY\_CLI\_TO\_SRV**  
The Encryption Key from client to server A single char of value 67 (ASCII char 'C').

**EVP\_KDF\_SSHKDF\_TYPE\_ENCRYPTION\_KEY\_SRV\_TO\_CLI**  
The Encryption Key from server to client A single char of value 68 (ASCII char 'D').

**EVP\_KDF\_SSHKDF\_TYPE\_INTEGRITY\_KEY\_CLI\_TO\_SRV**  
The Integrity Key from client to server A single char of value 69 (ASCII char 'E').

**EVP\_KDF\_SSHKDF\_TYPE\_INTEGRITY\_KEY\_SRV\_TO\_CLI**  
The Integrity Key from client to server A single char of value 70 (ASCII char 'F').

“constant” (**OSSL\_KDF\_PARAM\_CONSTANT**) <octet string>  
Sets the constant value in the associated KDF ctx.

“id” (**OSSL\_KDF\_PARAM\_PKCS12\_ID**) <integer>  
Sets the intended usage of the output bits in the associated KDF ctx. It is defined as per RFC 7292 section B.3.

## RETURN VALUES

**OSSL\_FUNC\_kdf\_newctx()** and **OSSL\_FUNC\_kdf\_dupctx()** should return the newly created provider side KDF context, or NULL on failure.

**OSSL\_FUNC\_kdf\_derive()**, **OSSL\_FUNC\_kdf\_get\_params()**, **OSSL\_FUNC\_kdf\_get\_ctx\_params()** and **OSSL\_FUNC\_kdf\_set\_ctx\_params()** should return 1 for success or 0 on error.

**OSSL\_FUNC\_kdf\_gettable\_params()**, **OSSL\_FUNC\_kdf\_gettable\_ctx\_params()** and **OSSL\_FUNC\_kdf\_settable\_ctx\_params()** should return a constant **OSSL\_PARAM** array, or NULL if none is offered.

## NOTES

The KDF life-cycle is described in **life\_cycle-kdf**(7). Providers should ensure that the various transitions listed there are supported. At some point the EVP layer will begin enforcing the listed transitions.

## SEE ALSO

**provider**(7), **life\_cycle-kdf**(7), **EVP\_KDF**(3).

**HISTORY**

The provider KDF interface was introduced in OpenSSL 3.0.

**COPYRIGHT**

Copyright 2020–2021 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>](https://www.openssl.org/source/license.html).