NAME

provider-mac - The mac 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_mac_newctx(void *provctx);
void OSSL_FUNC_mac_freectx(void *mctx);
void *OSSL_FUNC_mac_dupctx(void *src);

/* Encryption/decryption */
int OSSL_FUNC_mac_init(void *mctx, unsigned char *key, size_t keylen,
                       const OSSL_PARAM params[]);
int OSSL_FUNC_mac_update(void *mctx, const unsigned char *in, size_t inl);
int OSSL_FUNC_mac_final(void *mctx, unsigned char *out, size_t *outl, size_t outsize);

/* MAC parameter descriptors */
const OSSL_PARAM *OSSL_FUNC_mac_gettable_params(void *provctx);
const OSSL_PARAM *OSSL_FUNC_mac_gettable_ctx_params(void *mctx, void *provctx);
const OSSL_PARAM *OSSL_FUNC_mac_settable_ctx_params(void *mctx, void *provctx);

/* MAC parameters */
int OSSL_FUNC_mac_get_params(OSSL_PARAM params[]);
int OSSL_FUNC_mac_get_ctx_params(void *mctx, OSSL_PARAM params[]);
int OSSL_FUNC_mac_set_ctx_params(void *mctx, const OSSL_PARAM params[]);
```

DESCRIPTION

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

The MAC operation enables providers to implement mac algorithms and make them available to applications via the API functions **EVP_MAC_init**(3), **EVP_MAC_update**(3) and **EVP_MAC_final**(3).

All "functions" mentioned here are passed as function pointers between *libcrypto* and the provider in **OSSL_DISPATCH**(3) arrays via **OSSL_ALGORITHM**(3) 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**(3) element named **OSSL_FUNC_{name}**. For example, the "function" **OSSL_FUNC_mac_newctx()** has these:

```
typedef void *(OSSL_FUNC_mac_newctx_fn)(void *provctx);
static ossl_inline OSSL_FUNC_mac_newctx_fn
    OSSL_FUNC_mac_newctx(const OSSL_DISPATCH *opf);
```

OSSL_DISPATCH(3) arrays are indexed by numbers that are provided as macros in **openssl-core_dispatch.h**(7), as follows:

```
OSSL_FUNC_mac_newctx            OSSL_FUNC_MAC_NEWCTX
OSSL_FUNC_mac_freectx           OSSL_FUNC_MAC_FREECTX
OSSL_FUNC_mac_dupctx            OSSL_FUNC_MAC_DUPCTX

OSSL_FUNC_mac_init              OSSL_FUNC_MAC_INIT
OSSL_FUNC_mac_update            OSSL_FUNC_MAC_UPDATE
OSSL_FUNC_mac_final             OSSL_FUNC_MAC_FINAL

OSSL_FUNC_mac_get_params        OSSL_FUNC_MAC_GET_PARAMS
OSSL_FUNC_mac_get_ctx_params    OSSL_FUNC_MAC_GET_CTX_PARAMS
OSSL_FUNC_mac_set_ctx_params    OSSL_FUNC_MAC_SET_CTX_PARAMS

OSSL_FUNC_mac_gettable_params   OSSL_FUNC_MAC_GETTABLE_PARAMS
OSSL_FUNC_mac_gettable_ctx_params  OSSL_FUNC_MAC_GETTABLE_CTX_PARAMS
OSSL_FUNC_mac_settable_ctx_params  OSSL_FUNC_MAC_SETTABLE_CTX_PARAMS
```

A mac 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_mac_newctx()**, **OSSL_FUNC_mac_freectx()**, **OSSL_FUNC_mac_init()**, **OSSL_FUNC_mac_update()**, **OSSL_FUNC_mac_final()**. All other functions are optional.

Context Management Functions

OSSL_FUNC_mac_newctx() should create and return a pointer to a provider side structure for holding context information during a mac operation. A pointer to this context will be passed back in a number of the other mac operation function calls. The parameter *provctx* is the provider context generated

during provider initialisation (see **provider**(7)).

OSSL_FUNC_mac_freectx() is passed a pointer to the provider side mac context in the *mctx* parameter. If it receives NULL as *mctx* value, it should not do anything other than return. This function should free any resources associated with that context.

OSSL_FUNC_mac_dupctx() should duplicate the provider side mac context in the *mctx* parameter and return the duplicate copy.

Encryption/Decryption Functions

OSSL_FUNC_mac_init() initialises a mac operation given a newly created provider side mac context in the *mctx* parameter. The *params* are set before setting the MAC *key* of *keylen* bytes.

OSSL_FUNC_mac_update() is called to supply data for MAC computation of a previously initialised mac operation. The *mctx* parameter contains a pointer to a previously initialised provider side context. **OSSL_FUNC_mac_update()** may be called multiple times for a single mac operation.

OSSL_FUNC_mac_final() completes the MAC computation started through previous **OSSL_FUNC_mac_init()** and **OSSL_FUNC_mac_update()** calls. The *mctx* parameter contains a pointer to the provider side context. The resulting MAC should be written to *out* and the amount of data written to **outl*, which should not exceed *outsize* bytes. The same expectations apply to *outsize* as documented for **EVP_MAC_final**(3).

Mac Parameters

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

OSSL_FUNC_mac_get_params() gets details of parameter values associated with the provider algorithm and stores them in *params*.

OSSL_FUNC_mac_set_ctx_params() sets mac parameters associated with the given provider side mac context *mctx* to *params*. Any parameter settings are additional to any that were previously set. Passing NULL for *params* should return true.

OSSL_FUNC_mac_get_ctx_params() gets details of currently set parameter values associated with the given provider side mac context *mctx* and stores them in *params*. Passing NULL for *params* should return true.

OSSL_FUNC_mac_gettable_params(), **OSSL_FUNC_mac_gettable_ctx_params()**, and **OSSL_FUNC_mac_settable_ctx_params()** all return constant **OSSL_PARAM**(3) arrays as descriptors of the parameters that **OSSL_FUNC_mac_get_params()**, **OSSL_FUNC_mac_get_ctx_params()**, and

OSSL_FUNC_mac_set_ctx_params() can handle, respectively.
OSSL_FUNC_mac_gettable_ctx_params() and **OSSL_FUNC_mac_settable_ctx_params()** will return the parameters associated with the provider side context *mctx* in its current state if it is not NULL. Otherwise, they return the parameters associated with the provider side algorithm *provctx*.

All MAC implementations are expected to handle the following parameters:

with **OSSL_FUNC_set_ctx_params()**:

"key" (**OSSL_MAC_PARAM_KEY**) <octet string>

Sets the key in the associated MAC ctx. This is identical to passing a *key* argument to the **OSSL_FUNC_mac_init()** function.

with **OSSL_FUNC_get_params()**:

"size" (**OSSL_MAC_PARAM_SIZE**) <integer>

Can be used to get the default MAC size (which might be the only allowable MAC size for the implementation).

Note that some implementations allow setting the size that the resulting MAC should have as well, see the documentation of the implementation.

"size" (**OSSL_MAC_PARAM_BLOCK_SIZE**) <integer>

Can be used to get the MAC block size (if supported by the algorithm).

NOTES

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

RETURN VALUES

OSSL_FUNC_mac_newctx() and **OSSL_FUNC_mac_dupctx()** should return the newly created provider side mac context, or NULL on failure.

OSSL_FUNC_mac_init(), **OSSL_FUNC_mac_update()**, **OSSL_FUNC_mac_final()**, **OSSL_FUNC_mac_get_params()**, **OSSL_FUNC_mac_get_ctx_params()** and **OSSL_FUNC_mac_set_ctx_params()** should return 1 for success or 0 on error.

OSSL_FUNC_mac_gettable_params(), **OSSL_FUNC_mac_gettable_ctx_params()** and **OSSL_FUNC_mac_settable_ctx_params()** should return a constant **OSSL_PARAM**(3) array, or NULL if none is offered.

SEE ALSO

provider(7), **EVP_MAC-BLAKE2**(7), **EVP_MAC-CMAC**(7), **EVP_MAC-GMAC**(7), **EVP_MAC-HMAC**(7), **EVP_MAC-KMAC**(7), **EVP_MAC-Poly1305**(7), **EVP_MAC-Siphash**(7), **life_cycle-mac**(7), **EVP_MAC**(3)

HISTORY

The provider MAC interface was introduced in OpenSSL 3.0.

COPYRIGHT

Copyright 2019-2022 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>.