Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'provider-cipher.7ossl' command

$ man provider-cipher.7ossl

PROVIDER-CIPHER(7ossl) OpenSSL PROVIDER-CIPHER(7ossl)

NAME

provider-cipher - The cipher 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_cipher_newctx(void *provctx);
void OSSL_FUNC_cipher_freectx(void *cctx);
void *OSSL_FUNC_cipher_dupctx(void *cctx);
/* Encryption/decryption */
int OSSL_FUNC_cipher_encrypt_init(void *cctx, const unsigned char *key,
                                  size_t keylen, const unsigned char *iv,
                                  size_t ivlen, const OSSL_PARAM params[]);
int OSSL_FUNC_cipher_decrypt_init(void *cctx, const unsigned char *key,
                                  size_t keylen, const unsigned char *iv,
                                  size_t ivlen, const OSSL_PARAM params[]);
int OSSL_FUNC_cipher_update(void *cctx, unsigned char *out, size_t *outl,
```

```
                        size_t outsize, const unsigned char *in, size_t inl);
int OSSL_FUNC_cipher_final(void *cctx, unsigned char *out, size_t *outl,
                           size_t outsize);
int OSSL_FUNC_cipher_cipher(void *cctx, unsigned char *out, size_t *outl,
                            size_t outsize, const unsigned char *in, size_t inl);
/* Cipher parameter descriptors */
const OSSL_PARAM *OSSL_FUNC_cipher_gettable_params(void *provctx);
/* Cipher operation parameter descriptors */
const OSSL_PARAM *OSSL_FUNC_cipher_gettable_ctx_params(void *cctx,
                                                       void *provctx);
const OSSL_PARAM *OSSL_FUNC_cipher_settable_ctx_params(void *cctx,
                                                       void *provctx);
/* Cipher parameters */
int OSSL_FUNC_cipher_get_params(OSSL_PARAM params[]);
/* Cipher operation parameters */
int OSSL_FUNC_cipher_get_ctx_params(void *cctx, OSSL_PARAM params[]);
int OSSL_FUNC_cipher_set_ctx_params(void *cctx, const OSSL_PARAM params[]);
```

DESCRIPTION

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

The CIPHER operation enables providers to implement cipher algorithms and make them available to applications via the API functions EVP_EncryptInit_ex(3), EVP_EncryptUpdate(3) and EVP_EncryptFinal(3) (as well as the decrypt equivalents and other related functions).

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_cipher_newctx() has these:

```
typedef void *(OSSL_FUNC_cipher_newctx_fn)(void *provctx);
```

```
static ossl_inline OSSL_FUNC_cipher_newctx_fn
    OSSL_FUNC_cipher_newctx(const OSSL_DISPATCH *opf);
```

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

```
OSSL_FUNC_cipher_newctx             OSSL_FUNC_CIPHER_NEWCTX
OSSL_FUNC_cipher_freectx            OSSL_FUNC_CIPHER_FREECTX
OSSL_FUNC_cipher_dupctx             OSSL_FUNC_CIPHER_DUPCTX
OSSL_FUNC_cipher_encrypt_init       OSSL_FUNC_CIPHER_ENCRYPT_INIT
OSSL_FUNC_cipher_decrypt_init       OSSL_FUNC_CIPHER_DECRYPT_INIT
OSSL_FUNC_cipher_update             OSSL_FUNC_CIPHER_UPDATE
OSSL_FUNC_cipher_final              OSSL_FUNC_CIPHER_FINAL
OSSL_FUNC_cipher_cipher             OSSL_FUNC_CIPHER_CIPHER
OSSL_FUNC_cipher_get_params         OSSL_FUNC_CIPHER_GET_PARAMS
OSSL_FUNC_cipher_get_ctx_params     OSSL_FUNC_CIPHER_GET_CTX_PARAMS
OSSL_FUNC_cipher_set_ctx_params     OSSL_FUNC_CIPHER_SET_CTX_PARAMS
OSSL_FUNC_cipher_gettable_params    OSSL_FUNC_CIPHER_GETTABLE_PARAMS
OSSL_FUNC_cipher_gettable_ctx_params  OSSL_FUNC_CIPHER_GETTABLE_CTX_PARAMS
OSSL_FUNC_cipher_settable_ctx_params  OSSL_FUNC_CIPHER_SETTABLE_CTX_PARAMS
```

A cipher algorithm implementation may not implement all of these functions. In order to be a consistent set of functions there must at least be a complete set of "encrypt" functions, or a complete set of "decrypt" functions, or a single "cipher" function. In all cases both the OSSL_FUNC_cipher_newctx and OSSL_FUNC_cipher_freectx functions must be present. All other functions are optional.

Context Management Functions

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

OSSL_FUNC_cipher_freectx() is passed a pointer to the provider side cipher context in the cctx parameter. This function should free any

resources associated with that context.

OSSL_FUNC_cipher_dupctx() should duplicate the provider side cipher context in the cctx parameter and return the duplicate copy.

Encryption/Decryption Functions

OSSL_FUNC_cipher_encrypt_init() initialises a cipher operation for encryption given a newly created provider side cipher context in the cctx parameter. The key to be used is given in key which is keylen bytes long. The IV to be used is given in iv which is ivlen bytes long. The params, if not NULL, should be set on the context in a manner similar to using OSSL_FUNC_cipher_set_ctx_params().

OSSL_FUNC_cipher_decrypt_init() is the same as OSSL_FUNC_cipher_encrypt_init() except that it initialises the context for a decryption operation.

OSSL_FUNC_cipher_update() is called to supply data to be encrypted/decrypted as part of a previously initialised cipher operation. The cctx parameter contains a pointer to a previously initialised provider side context. OSSL_FUNC_cipher_update() should encrypt/decrypt inl bytes of data at the location pointed to by in. The encrypted data should be stored in out and the amount of data written to *outl which should not exceed outsize bytes.

OSSL_FUNC_cipher_update() may be called multiple times for a single cipher operation. It is the responsibility of the cipher implementation to handle input lengths that are not multiples of the block length. In such cases a cipher implementation will typically cache partial blocks of input data until a complete block is obtained. out may be the same location as in but it should not partially overlap. The same expectations apply to outsize as documented for EVP_EncryptUpdate(3) and EVP_DecryptUpdate(3).

OSSL_FUNC_cipher_final() completes an encryption or decryption started through previous OSSL_FUNC_cipher_encrypt_init() or OSSL_FUNC_cipher_decrypt_init(), and OSSL_FUNC_cipher_update() calls. The cctx parameter contains a pointer to the provider side context. Any final encryption/decryption output 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_EncryptFinal(3) and EVP_DecryptFinal(3).
OSSL_FUNC_cipher_cipher() performs encryption/decryption using the
provider side cipher context in the cctx parameter that should have
been previously initialised via a call to
OSSL_FUNC_cipher_encrypt_init() or OSSL_FUNC_cipher_decrypt_init().
This should call the raw underlying cipher function without any
padding. This will be invoked in the provider as a result of the
application calling EVP_Cipher(3). The application is responsible for
ensuring that the input is a multiple of the block length. The data to
be encrypted/decrypted will be in in, and it will be inl bytes in
length. The output from the encryption/decryption should be stored in
out and the amount of data stored should be put in *outl which should
be no more than outsize bytes.

Cipher Parameters

See OSSL_PARAM(3) for further details on the parameters structure used
by these functions.
OSSL_FUNC_cipher_get_params() gets details of the algorithm
implementation and stores them in params.
OSSL_FUNC_cipher_set_ctx_params() sets cipher operation parameters for
the provider side cipher context cctx to params. Any parameter
settings are additional to any that were previously set. Passing NULL
for params should return true.
OSSL_FUNC_cipher_get_ctx_params() gets cipher operation details details
from the given provider side cipher context cctx and stores them in
params. Passing NULL for params should return true.
OSSL_FUNC_cipher_gettable_params(),
OSSL_FUNC_cipher_gettable_ctx_params(), and
OSSL_FUNC_cipher_settable_ctx_params() all return constant OSSL_PARAM
arrays as descriptors of the parameters that
OSSL_FUNC_cipher_get_params(), OSSL_FUNC_cipher_get_ctx_params(), and
OSSL_FUNC_cipher_set_ctx_params() can handle, respectively.

OSSL_FUNC_cipher_gettable_ctx_params() and
OSSL_FUNC_cipher_settable_ctx_params() will return the parameters
associated with the provider side context cctx 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 ciphers are listed in
"PARAMETERS" in EVP_EncryptInit(3). Not all parameters are relevant
to, or are understood by all ciphers.

RETURN VALUES

OSSL_FUNC_cipher_newctx() and OSSL_FUNC_cipher_dupctx() should return
the newly created provider side cipher context, or NULL on failure.
OSSL_FUNC_cipher_encrypt_init(), OSSL_FUNC_cipher_decrypt_init(),
OSSL_FUNC_cipher_update(), OSSL_FUNC_cipher_final(),
OSSL_FUNC_cipher_cipher(), OSSL_FUNC_cipher_get_params(),
OSSL_FUNC_cipher_get_ctx_params() and OSSL_FUNC_cipher_set_ctx_params()
should return 1 for success or 0 on error.
OSSL_FUNC_cipher_gettable_params(),
OSSL_FUNC_cipher_gettable_ctx_params() and
OSSL_FUNC_cipher_settable_ctx_params() should return a constant
OSSL_PARAM array, or NULL if none is offered.

SEE ALSO

provider(7), OSSL_PROVIDER-FIPS(7), OSSL_PROVIDER-default(7),
OSSL_PROVIDER-legacy(7), EVP_CIPHER-AES(7), EVP_CIPHER-ARIA(7),
EVP_CIPHER-BLOWFISH(7), EVP_CIPHER-CAMELLIA(7), EVP_CIPHER-CAST(7),
EVP_CIPHER-CHACHA(7), EVP_CIPHER-DES(7), EVP_CIPHER-IDEA(7),
EVP_CIPHER-RC2(7), EVP_CIPHER-RC4(7), EVP_CIPHER-RC5(7),
EVP_CIPHER-SEED(7), EVP_CIPHER-SM4(7), life_cycle-cipher(7),
EVP_EncryptInit(3)

HISTORY

The provider CIPHER interface was introduced in OpenSSL 3.0.

COPYRIGHT

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>.