NAME

HMAC, HMAC_CTX_new, HMAC_CTX_reset, HMAC_CTX_free, HMAC_Init, HMAC_Init_ex, HMAC_Update, HMAC_Final, HMAC_CTX_copy, HMAC_CTX_set_flags, HMAC_CTX_get_md, HMAC_size - HMAC message authentication code

SYNOPSIS

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

unsigned char *HMAC(const EVP_MD *evp_md, const void *key, int key_len,
                    const unsigned char *data, size_t data_len,
                    unsigned char *md, unsigned int *md_len);
```

The following functions have been deprecated since OpenSSL 3.0, and can be hidden entirely by defining **OPENSSL_API_COMPAT** with a suitable version value, see **openssl_user_macros**(7):

```
HMAC_CTX *HMAC_CTX_new(void);
int HMAC_CTX_reset(HMAC_CTX *ctx);

int HMAC_Init_ex(HMAC_CTX *ctx, const void *key, int key_len,
                 const EVP_MD *md, ENGINE *impl);
int HMAC_Update(HMAC_CTX *ctx, const unsigned char *data, size_t len);
int HMAC_Final(HMAC_CTX *ctx, unsigned char *md, unsigned int *len);

void HMAC_CTX_free(HMAC_CTX *ctx);

int HMAC_CTX_copy(HMAC_CTX *dctx, HMAC_CTX *sctx);
void HMAC_CTX_set_flags(HMAC_CTX *ctx, unsigned long flags);
const EVP_MD *HMAC_CTX_get_md(const HMAC_CTX *ctx);

size_t HMAC_size(const HMAC_CTX *e);
```

The following function has been deprecated since OpenSSL 1.1.0, and can be hidden entirely by defining **OPENSSL_API_COMPAT** with a suitable version value, see **openssl_user_macros**(7):

```
int HMAC_Init(HMAC_CTX *ctx, const void *key, int key_len,
              const EVP_MD *md);
```

DESCRIPTION

HMAC is a MAC (message authentication code), i.e. a keyed hash function used for message authentication, which is based on a hash function.

HMAC() computes the message authentication code of the *data_len* bytes at *data* using the hash function *evp_md* and the key *key* which is *key_len* bytes long. The *key* may also be NULL with *key_len* being 0.

It places the result in *md* (which must have space for the output of the hash function, which is no more than **EVP_MAX_MD_SIZE** bytes). If *md* is NULL, the digest is placed in a static array. The size of the output is placed in *md_len*, unless it is NULL. Note: passing a NULL value for *md* to use the static array is not thread safe.

evp_md is a message digest such as **EVP_sha1()**, **EVP_ripemd160()** etc. HMAC does not support variable output length digests such as **EVP_shake128()** and **EVP_shake256()**.

HMAC() uses the default **OSSL_LIB_CTX**. Use **EVP_Q_mac**(3) instead if a library context is required.

All of the functions described below are deprecated. Applications should instead use **EVP_MAC_CTX_new**(3), **EVP_MAC_CTX_free**(3), **EVP_MAC_init**(3), **EVP_MAC_update**(3) and **EVP_MAC_final**(3) or the 'quick' single-shot MAC function **EVP_Q_mac**(3).

HMAC_CTX_new() creates a new HMAC_CTX in heap memory.

HMAC_CTX_reset() clears an existing **HMAC_CTX** and associated resources, making it suitable for new computations as if it was newly created with **HMAC_CTX_new()**.

HMAC_CTX_free() erases the key and other data from the **HMAC_CTX**, releases any associated resources and finally frees the **HMAC_CTX** itself.

The following functions may be used if the message is not completely stored in memory:

HMAC_Init_ex() initializes or reuses a **HMAC_CTX** structure to use the hash function *evp_md* and key *key*. If both are NULL, or if *key* is NULL and *evp_md* is the same as the previous call, then the existing key is reused. *ctx* must have been created with **HMAC_CTX_new()** before the first use of an **HMAC_CTX** in this function.

If **HMAC_Init_ex()** is called with *key* NULL and *evp_md* is not the same as the previous digest used by *ctx* then an error is returned because reuse of an existing key with a different digest is not supported.

HMAC_Init() initializes a **HMAC_CTX** structure to use the hash function *evp_md* and the key *key* which is *key_len* bytes long.

HMAC_Update() can be called repeatedly with chunks of the message to be authenticated (*len* bytes at *data*).

HMAC_Final() places the message authentication code in *md*, which must have space for the hash function output.

HMAC_CTX_copy() copies all of the internal state from *sctx* into *dctx*.

HMAC_CTX_set_flags() applies the specified flags to the internal EVP_MD_CTXs. These flags have the same meaning as for **EVP_MD_CTX_set_flags**(3).

HMAC_CTX_get_md() returns the EVP_MD that has previously been set for the supplied HMAC_CTX.

HMAC_size() returns the length in bytes of the underlying hash function output.

RETURN VALUES

HMAC() returns a pointer to the message authentication code or NULL if an error occurred.

HMAC_CTX_new() returns a pointer to a new **HMAC_CTX** on success or NULL if an error occurred.

HMAC_CTX_reset(), **HMAC_Init_ex()**, **HMAC_Update()**, **HMAC_Final()** and **HMAC_CTX_copy()** return 1 for success or 0 if an error occurred.

HMAC_CTX_get_md() return the EVP_MD previously set for the supplied HMAC_CTX or NULL if no EVP_MD has been set.

HMAC_size() returns the length in bytes of the underlying hash function output or zero on error.

CONFORMING TO

RFC 2104

SEE ALSO

SHA1(3), **EVP_Q_mac**(3), **evp**(7)

HISTORY

All functions except for **HMAC()** were deprecated in OpenSSL 3.0.

HMAC_CTX_init() was replaced with **HMAC_CTX_reset()** in OpenSSL 1.1.0.

HMAC_CTX_cleanup() existed in OpenSSL before version 1.1.0.

HMAC_CTX_new(), **HMAC_CTX_free()** and **HMAC_CTX_get_md()** are new in OpenSSL 1.1.0.

HMAC_Init_ex(), **HMAC_Update()** and **HMAC_Final()** did not return values in OpenSSL before version 1.0.0.

COPYRIGHT