NAME

EVP_MAC−KMAC, EVP_MAC−KMAC128, EVP_MAC−KMAC256 − The KMAC EVP_MAC implementations

DESCRIPTION

Support for computing KMAC MACs through the **EVP_MAC** API.

Identity

These implementations are identified with one of these names and properties, to be used with **EVP_MAC_fetch()**:

‘‘KMAC−128’’, ‘‘provider=default’’ or ‘‘provider=fips’’
‘‘KMAC−256’’, ‘‘provider=default’’ or ‘‘provider=fips’’

Supported parameters

The general description of these parameters can be found in ‘‘PARAMETERS’’ in **EVP_MAC** (3).

All these parameters can be set with **EVP_MAC_CTX_set_params()**. Furthermore, the ‘‘size’’ parameter can be retrieved with **EVP_MAC_CTX_get_params()**, or with **EVP_MAC_CTX_get_mac_size()**. The length of the ‘‘size’’ parameter should not exceed that of a **size_t**. Likewise, the ‘‘block-size’’ parameter can be retrieved with **EVP_MAC_CTX_get_params()**, or with **EVP_MAC_CTX_get_block_size()**.

‘‘key’’ (**OSSL_MAC_PARAM_KEY**) <octet string>
Sets the MAC key. Setting this parameter is identical to passing a*k ey* to **EVP_MAC_init** (3).

‘‘custom’’ (**OSSL_MAC_PARAM_CUSTOM**) <octet string>
Sets the custom value. It is an optional value of at most 256 bytes, and is empty by default.

‘‘size’’ (**OSSL_MAC_PARAM_SIZE**) <unsigned integer>
Sets the MAC size. By default, it is 16 for `KMAC−128` and 32 for `KMAC−256`.

‘‘block-size’’ (**OSSL_MAC_PARAM_SIZE**) <unsigned integer>
Gets the MAC block size. By default, it is 168 for `KMAC−128` and 136 for `KMAC−256`.

‘‘xof’’ (**OSSL_MAC_PARAM_XOF**) <integer>
The ‘‘xof’’ parameter value is expected to be 1 or 0. Use 1 to enable XOF mode. The default value is 0.

The ‘‘custom’’ parameter must be set as part of or before the **EVP_MAC_init()** call. The ‘‘xof’’ and ‘‘size’’ parameters can be set at any time before **EVP_MAC_final()**. The ‘‘key’’ parameter is set as part of the **EVP_MAC_init()** call, but can be set before it instead.

EXAMPLES

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

static int do_kmac(const unsigned char *in, size_t in_len,
                   const unsigned char *key, size_t key_len,
                   const unsigned char *custom, size_t custom_len,
                   int xof_enabled, unsigned char *out, int out_len)
{
    EVP_MAC_CTX *ctx = NULL;
    EVP_MAC *mac = NULL;
    OSSL_PARAM params[4], *p;
    int ret = 0;
    size_t l = 0;

    mac = EVP_MAC_fetch(NULL, "KMAC−128", NULL);
    if (mac == NULL)
        goto err;
    ctx = EVP_MAC_CTX_new(mac);
    /* The mac can be freed after it is used by EVP_MAC_CTX_new */
    EVP_MAC_free(mac);
    if (ctx == NULL)
        goto err;

    /*
     * Setup parameters required before calling EVP_MAC_init()
     * The parameters OSSL_MAC_PARAM_XOF and OSSL_MAC_PARAM_SIZE may also be
     * used at this point.
     */
    p = params;
    *p++ = OSSL_PARAM_construct_octet_string(OSSL_MAC_PARAM_KEY,
                                             (void *)key, key_len);
    if (custom != NULL && custom_len != 0)
      *p++ = OSSL_PARAM_construct_octet_string(OSSL_MAC_PARAM_CUSTOM,
                                               (void *)custom, custom_len);
    *p = OSSL_PARAM_construct_end();
    if (!EVP_MAC_CTX_set_params(ctx, params))
        goto err;

    if (!EVP_MAC_init(ctx))
        goto err;

    /*
     * Note: the following optional parameters can be set any time
     * before EVP_MAC_final().
     */
    p = params;
    *p++ = OSSL_PARAM_construct_int(OSSL_MAC_PARAM_XOF, &xof_enabled);
    *p++ = OSSL_PARAM_construct_int(OSSL_MAC_PARAM_SIZE, &out_len);
    *p = OSSL_PARAM_construct_end();
    if (!EVP_MAC_CTX_set_params(ctx, params))
        goto err;

    /* The update may be called multiple times here for streamed input */
    if (!EVP_MAC_update(ctx, in, in_len))
        goto err;
    if (!EVP_MAC_final(ctx, out, &l, out_len))
        goto err;
    ret = 1;
err:
    EVP_MAC_CTX_free(ctx);
    return ret;
}
```

SEE ALSO

EVP_MAC_CTX_get_params (3), **EVP_MAC_CTX_set_params** (3), "PARAMETERS" in **EVP_MAC** (3), **OSSL_PARAM** (3)

COPYRIGHT