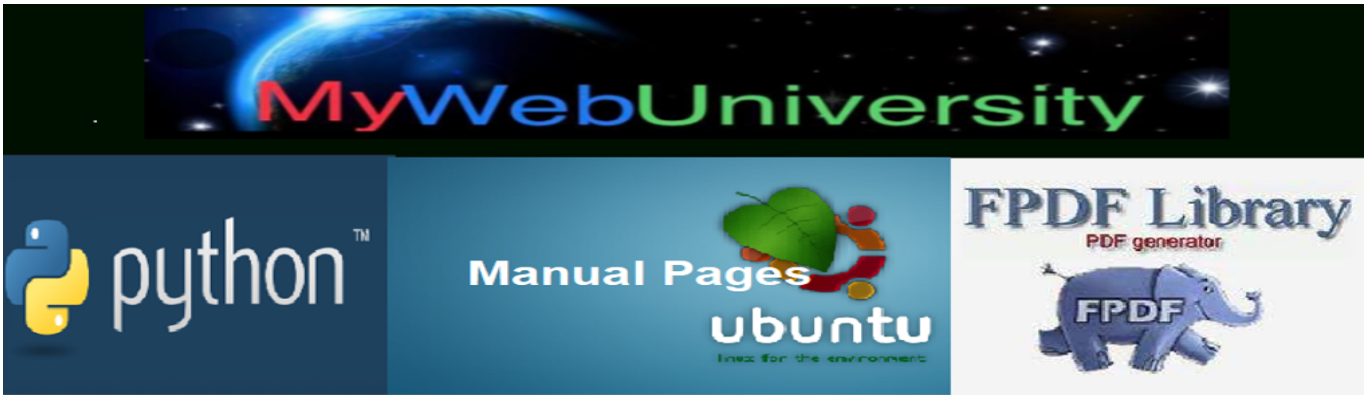

Rocky Enterprise Linux 9.2 Manual Pages on command 'EVP_KDF-HKDF.7ssl'

$ man EVP_KDF-HKDF.7ssl

NAME

EVP_KDF-HKDF - The HKDF EVP_KDF implementation

DESCRIPTION

Support for computing the HKDF KDF through the EVP_KDF API.

The EVP_KDF-HKDF algorithm implements the HKDF key derivation function. HKDF follows the "extract-then-expand" paradigm, where the KDF logically consists of two modules. The first stage takes the input keying material and "extracts" from it a fixed-length pseudorandom key K. The second stage "expands" the key K into several additional pseudorandom keys (the output of the KDF).

Identity

"HKDF" is the name for this implementation; it can be used with the EVP_KDF_fetch() function.

Supported parameters

The supported parameters are:

"properties" (OSSL_KDF_PARAM_PROPERTIES) <UTF8 string>

"digest" (OSSL_KDF_PARAM_DIGEST) <UTF8 string>

"key" (OSSL_KDF_PARAM_KEY) <octet string>

"salt" (OSSL_KDF_PARAM_SALT) <octet string>

These parameters work as described in "PARAMETERS" in EVP_KDF(3).

"info" (OSSL_KDF_PARAM_INFO) <octet string>

This parameter sets the info value. The length of the context info buffer cannot exceed 1024 bytes; this should be more than enough for any normal use of HKDF.

"mode" (OSSL_KDF_PARAM_MODE) <UTF8 string> or <integer>

This parameter sets the mode for the HKDF operation. There are three modes that are currently defined:

"EXTRACT_AND_EXPAND" or EVP_KDF_HKDF_MODE_EXTRACT_AND_EXPAND

This is the default mode. Calling EVP_KDF_derive(3) on an EVP_KDF_CTX set up for HKDF will perform an extract followed by an expand operation in one go. The derived key returned will be the result after the expand operation. The intermediate fixed-length pseudorandom key K is not returned.

In this mode the digest, key, salt and info values must be set before a key is derived otherwise an error will occur.

"EXTRACT_ONLY" or EVP_KDF_HKDF_MODE_EXTRACT_ONLY

In this mode calling EVP_KDF_derive(3) will just perform the extract operation. The value returned will be the intermediate fixed-length pseudorandom key K. The keylen parameter must match the size of K, which can be looked up by calling EVP_KDF_CTX_get_kdf_size() after setting the mode and digest.

The digest, key and salt values must be set before a key is derived otherwise an error will occur.

"EXPAND_ONLY" or EVP_KDF_HKDF_MODE_EXPAND_ONLY

In this mode calling EVP_KDF_derive(3) will just perform the expand operation. The input key should be set to the intermediate fixed-length pseudorandom key K

returned from a previous extract operation.

The digest, key and info values must be set before a key is derived otherwise an error will occur.

NOTES

A context for HKDF can be obtained by calling:

```
EVP_KDF *kdf = EVP_KDF_fetch(NULL, "HKDF", NULL);
EVP_KDF_CTX *kctx = EVP_KDF_CTX_new(kdf);
```

The output length of an HKDF expand operation is specified via the keylen parameter to the EVP_KDF_derive(3) function. When using EVP_KDF_HKDF_MODE_EXTRACT_ONLY the keylen parameter must equal the size of the intermediate fixed-length pseudorandom key otherwise an error will occur. For that mode, the fixed output size can be looked up by calling EVP_KDF_CTX_get_kdf_size() after setting the mode and digest on the EVP_KDF_CTX.

EXAMPLES

This example derives 10 bytes using SHA-256 with the secret key "secret", salt value "salt" and info value "label":

```
EVP_KDF *kdf;
EVP_KDF_CTX *kctx;
unsigned char out[10];
OSSL_PARAM params[5], *p = params;

kdf = EVP_KDF_fetch(NULL, "HKDF", NULL);
kctx = EVP_KDF_CTX_new(kdf);
EVP_KDF_free(kdf);

*p++ = OSSL_PARAM_construct_utf8_string(OSSL_KDF_PARAM_DIGEST,
                                        SN_sha256, strlen(SN_sha256));
*p++ = OSSL_PARAM_construct_octet_string(OSSL_KDF_PARAM_KEY,
```

```
                                        "secret", (size_t)6);
    *p++ = OSSL_PARAM_construct_octet_string(OSSL_KDF_PARAM_INFO,
                                        "label", (size_t)5);
    *p++ = OSSL_PARAM_construct_octet_string(OSSL_KDF_PARAM_SALT,
                                        "salt", (size_t)4);
    *p = OSSL_PARAM_construct_end();
    if (EVP_KDF_derive(kctx, out, sizeof(out), params) <= 0) {
        error("EVP_KDF_derive");
    }

    EVP_KDF_CTX_free(kctx);
```

CONFORMING TO

RFC 5869

SEE ALSO

EVP_KDF(3), EVP_KDF_CTX_new(3), EVP_KDF_CTX_free(3), EVP_KDF_CTX_get_kdf_size(3), EVP_KDF_CTX_set_params(3), EVP_KDF_derive(3), "PARAMETERS" in EVP_KDF(3), EVP_KDF-TLS13_KDF(7)

COPYRIGHT