



## **Red Hat Enterprise Linux Release 9.2 Manual Pages on 'EVP\_KDF-SS.7ossl' command**

**\$ man EVP\_KDF-SS.7ossl**

EVP\_KDF-SS(7ossl)                  OpenSSL                  EVP\_KDF-SS(7ossl)

### NAME

EVP\_KDF-SS - The Single Step / One Step EVP\_KDF implementation

### DESCRIPTION

The EVP\_KDF-SS algorithm implements the Single Step key derivation function (SSKDF). SSKDF derives a key using input such as a shared secret key (that was generated during the execution of a key establishment scheme) and fixedinfo. SSKDF is also informally referred to as 'Concat KDF'.

### Auxiliary function

The implementation uses a selectable auxiliary function H, which can be one of:

$H(x) = \text{hash}(x, \text{digest}=\text{md})$

$H(x) = \text{HMAC\_hash}(x, \text{key}=\text{salt}, \text{digest}=\text{md})$

$H(x) = \text{KMACxxx}(x, \text{key}=\text{salt}, \text{custom}=\text{"KDF"}, \text{outlen}=\text{mac\_size})$

Both the HMAC and KMAC implementations set the key using the 'salt' value. The hash and HMAC also require the digest to be set.

## Identity

"SSKDF" 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>

"mac" (OSSL\_KDF\_PARAM\_MAC) <UTF8 string>

"maclen" (OSSL\_KDF\_PARAM\_MAC\_SIZE) <unsigned integer>

"salt" (OSSL\_KDF\_PARAM\_SALT) <octet string>

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

"key" (EVP\_KDF\_CTRL\_SET\_KEY) <octet string>

This parameter set the shared secret that is used for key derivation.

"info" (OSSL\_KDF\_PARAM\_INFO) <octet string>

This parameter sets an optional value for fixedinfo, also known as otherinfo.

## NOTES

A context for SSKDF can be obtained by calling:

```
EVP_KDF *kdf = EVP_KDF_fetch(NULL, "SSKDF", NULL);
```

```
EVP_KDF_CTX *kctx = EVP_KDF_CTX_new(kdf);
```

The output length of an SSKDF is specified via the `keylen` parameter to the `EVP_KDF_derive(3)` function.

## EXAMPLES

This example derives 10 bytes using  $H(x) = \text{SHA-256}$ , with the secret key

"secret" and fixedinfo value "label":

```
EVP_KDF *kdf;
EVP_KDF_CTX *kctx;
unsigned char out[10];
OSSL_PARAM params[4], *p = params;

kdf = EVP_KDF_fetch(NULL, "SSKDF", 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_end();
if (EVP_KDF_derive(kctx, out, sizeof(out), params) <= 0) {
    error("EVP_KDF_derive");
}

EVP_KDF_CTX_free(kctx);
```

This example derives 10 bytes using  $H(x) = \text{HMAC}(\text{SHA-256})$ , with the secret key "secret", fixedinfo value "label" and salt "salt":

```
EVP_KDF *kdf;
EVP_KDF_CTX *kctx;
unsigned char out[10];
OSSL_PARAM params[6], *p = params;

kdf = EVP_KDF_fetch(NULL, "SSKDF", NULL);
```

```

kctx = EVP_KDF_CTX_new(kdf);
EVP_KDF_free(kdf);

*p++ = OSSL_PARAM_construct_utf8_string(OSSL_KDF_PARAM_MAC,
                                         SN_hmac, strlen(SN_hmac));

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

*p++ = OSSL_PARAM_construct_octet_string(EVP_KDF_CTRL_SET_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);

```

This example derives 10 bytes using  $H(x) = \text{KMAC128}(x, \text{salt}, \text{outlen})$ , with the secret key "secret" fixed info value "label", salt of "salt" and KMAC outlen of 20:

```

EVP_KDF *kdf;
EVP_KDF_CTX *kctx;
unsigned char out[10];
OSSL_PARAM params[7], *p = params;

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

```

```

*p++ = OSSL_PARAM_construct_utf8_string(OSSL_KDF_PARAM_MAC,
                                         SN_kmac128, strlen(SN_kmac128));
*p++ = OSSL_PARAM_construct_utf8_string(OSSL_KDF_PARAM_DIGEST,
                                         SN_sha256, strlen(SN_sha256));
*p++ = OSSL_PARAM_construct_octet_string(EVP_KDF_CTRL_SET_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_size_t(OSSL_KDF_PARAM_MAC_SIZE, (size_t)20);
*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

NIST SP800-56Cr1.

## SEE ALSO

EVP\_KDF(3), EVP\_KDF\_CTX\_new(3), EVP\_KDF\_CTX\_free(3),  
EVP\_KDF\_CTX\_set\_params(3), EVP\_KDF\_CTX\_get\_kdf\_size(3),  
EVP\_KDF\_derive(3), "PARAMETERS" in EVP\_KDF(3)

## HISTORY

This functionality was added to OpenSSL 3.0.

## COPYRIGHT

Copyright 2019-2021 The OpenSSL Project Authors. All Rights Reserved.

Copyright (c) 2019, Oracle and/or its affiliates. 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>.

3.0.7

2023-07-13

EVP\_KDF-SS(7ossl)