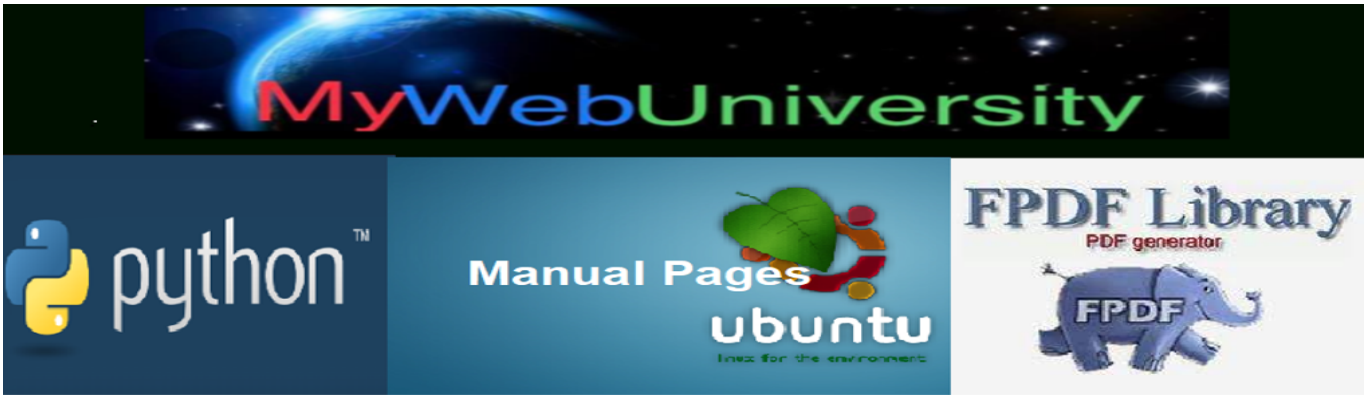

Rocky Enterprise Linux 9.2 Manual Pages on command 'Crypt::KeyDerivation.3pm'

$ man Crypt::KeyDerivation.3pm

NAME

Crypt::KeyDerivation - PBKDF1, PBKDF2 and HKDF key derivation functions

SYNOPSIS

```
use Crypt::KeyDerivation ':all';

### PBKDF1/2
$derived_key1 = pbkdf1($password, $salt, $iteration_count, $hash_name, $len);
$derived_key2 = pbkdf2($password, $salt, $iteration_count, $hash_name, $len);

### HKDF & co.
$derived_key3 = hkdf($keying_material, $salt, $hash_name, $len, $info);
$prk  = hkdf_extract($keying_material, $salt, $hash_name);
$okm1 = hkdf_expand($prk, $hash_name, $len, $info);
```

DESCRIPTION

Provides an interface to Key derivation functions:

? PBKDF1 and PBKDF according to PKCS#5 v2.0 <https://tools.ietf.org/html/rfc2898>

? HKDF (+ related) according to <https://tools.ietf.org/html/rfc5869>

FUNCTIONS

pbkdf1

BEWARE: if you are not sure, do not use "pbkdf1" but rather choose "pbkdf2".

```
$derived_key = pbkdf1($password, $salt, $iteration_count, $hash_name, $len);
#or
$derived_key = pbkdf1($password, $salt, $iteration_count, $hash_name);
#or
$derived_key = pbkdf1($password, $salt, $iteration_count);
#or
$derived_key = pbkdf1($password, $salt);

# $password ..... input keying material  (password)
# $salt ..... salt/nonce (expected length: 8)
# $iteration_count .. optional, DEFAULT: 5000
# $hash_name ..... optional, DEFAULT: 'SHA256'
# $len ..... optional, derived key len, DEFAULT: 32
```

pbkdf2

```
$derived_key = pbkdf2($password, $salt, $iteration_count, $hash_name, $len);
#or
$derived_key = pbkdf2($password, $salt, $iteration_count, $hash_name);
#or
$derived_key = pbkdf2($password, $salt, $iteration_count);
#or
$derived_key = pbkdf2($password, $salt);

# $password ..... input keying material (password)
# $salt ..... salt/nonce
# $iteration_count .. optional, DEFAULT: 5000
# $hash_name ..... optional, DEFAULT: 'SHA256'
# $len ..... optional, derived key len, DEFAULT: 32
```

hkdf

```
$okm2 = hkdf($password, $salt, $hash_name, $len, $info);
#or
$okm2 = hkdf($password, $salt, $hash_name, $len);
#or
$okm2 = hkdf($password, $salt, $hash_name);
#or
$okm2 = hkdf($password, $salt);

# $password ... input keying material (password)
# $salt ..... salt/nonce, if undef defaults to HashLen zero octets
# $hash_name .. optional, DEFAULT: 'SHA256'
# $len ..... optional, derived key len, DEFAULT: 32
# $info ..... optional context and application specific information, DEFAULT: ''
```

hkdf_extract

```
$prk  = hkdf_extract($password, $salt, $hash_name);
#or
$prk  = hkdf_extract($password, $salt, $hash_name);

# $password ... input keying material (password)
# $salt ..... salt/nonce, if undef defaults to HashLen zero octets
# $hash_name .. optional, DEFAULT: 'SHA256'
```

hkdf_expand

```
$okm = hkdf_expand($pseudokey, $hash_name, $len, $info);
#or
$okm = hkdf_expand($pseudokey, $hash_name, $len);
#or
$okm = hkdf_expand($pseudokey, $hash_name);
#or
$okm = hkdf_expand($pseudokey);
```

```
    # $pseudokey .. input keying material
    # $hash_name .. optional, DEFAULT: 'SHA256'
    # $len ..... optional, derived key len, DEFAULT: 32
    # $info ..... optional context and application specific information, DEFAULT: ''
```