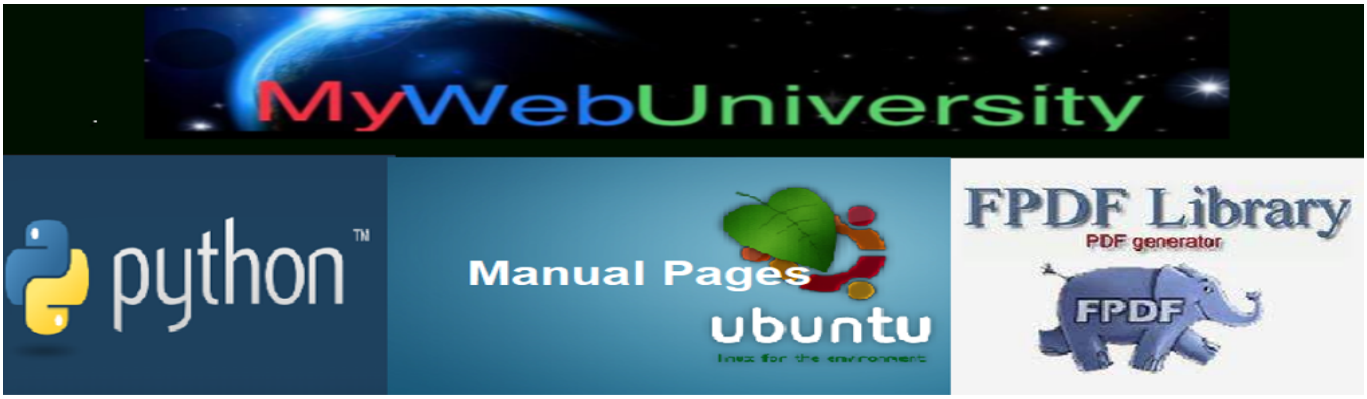

Rocky Enterprise Linux 9.2 Manual Pages on command 'Crypt::Cipher.3pm'

$ man Crypt::Cipher.3pm

Crypt::Cipher(3pm) User Contributed Perl Documentation Crypt::Cipher(3pm)

NAME

Crypt::Cipher - Generic interface to cipher functions

SYNOPSIS

```
#### example 1 (encrypting single block)
use Crypt::Cipher;

my $key = '...'; # length has to be valid key size for this cipher
my $c = Crypt::Cipher->new('AES', $key);
my $blocksize  = $c->blocksize;
my $ciphertext = $c->encrypt('plain text block'); #encrypt 1 block
my $plaintext  = $c->decrypt($ciphertext);        #decrypt 1 block

### example 2 (using CBC mode)
use Crypt::Mode::CBC;

my $key = '...'; # length has to be valid key size for this cipher
my $iv = '...';  # 16 bytes
my $cbc = Crypt::Mode::CBC->new('AES');
my $ciphertext = $cbc->encrypt("secret data", $key, $iv);
```

```
#### example 3 (compatibility with Crypt::CBC)
use Crypt::CBC;
use Crypt::Cipher;

my $key = '...'; # length has to be valid key size for this cipher
my $iv = '...';  # 16 bytes
my $cipher = Crypt::Cipher('AES', $key);
my $cbc = Crypt::CBC->new( -cipher=>$cipher, -iv=>$iv );
my $ciphertext = $cbc->encrypt("secret data");
```

DESCRIPTION

Provides an interface to various symmetric cipher algorithms.

BEWARE: This module implements just elementary "one-block-(en|de)cryption" operation - if you want to encrypt/decrypt generic data you have to use some of the cipher block modes - check for example Crypt::Mode::CBC, Crypt::Mode::CTR or Crypt::CBC (which will be slower).

METHODS

new

Constructor, returns a reference to the cipher object.

```
## basic scenario
$d = Crypt::Cipher->new($name, $key);
# $name = one of 'AES', 'Anubis', 'Blowfish', 'CAST5', 'Camellia', 'DES', 'DES_EDE',
#             'KASUMI', 'Khazad', 'MULTI2', 'Noekeon', 'RC2', 'RC5', 'RC6',
#             'SAFERP', 'SAFER_K128', 'SAFER_K64', 'SAFER_SK128', 'SAFER_SK64',
#             'SEED', 'Skipjack', 'Twofish', 'XTEA', 'IDEA', 'Serpent'
#             simply any <NAME> for which there exists Crypt::Cipher::<NAME>
# $key = binary key (keysize should comply with selected cipher requirements)

## some of the ciphers (e.g. MULTI2, RC5, SAFER) allow one to set number of rounds
$d = Crypt::Cipher->new('MULTI2', $key, $rounds);
# $rounds = positive integer (should comply with selected cipher requirements)
```

encrypt

Encrypts $plaintext and returns the $ciphertext where $plaintext and $ciphertext should be of blocksize bytes.

```
$ciphertext = $d->encrypt($plaintext);
```

decrypt

Decrypts $ciphertext and returns the $plaintext where $plaintext and $ciphertext should be of blocksize bytes.

```
$plaintext = $d->decrypt($ciphertext);
```

keysize

Just an alias for max_keysize (needed for Crypt::CBC compatibility).

max_keysize

Returns the maximal allowed key size (in bytes) for given cipher.

```
$d->max_keysize;
#or
Crypt::Cipher->max_keysize('AES');
#or
Crypt::Cipher::max_keysize('AES');
```

min_keysize

Returns the minimal allowed key size (in bytes) for given cipher.

```
$d->min_keysize;
#or
Crypt::Cipher->min_keysize('AES');
#or
Crypt::Cipher::min_keysize('AES');
```

blocksize

Returns block size (in bytes) for given cipher.

```
$d->blocksize;
#or
Crypt::Cipher->blocksize('AES');
#or
Crypt::Cipher::blocksize('AES');
```

default_rounds

Returns default number of rounds for given cipher. NOTE: only some ciphers (e.g. MULTI2, RC5, SAFER) allow one to set number of rounds via new().

```
$d->default_rounds;
#or
Crypt::Cipher->default_rounds('AES');
#or
Crypt::Cipher::default_rounds('AES');
```

SEE ALSO

- CryptX
- Check subclasses like Crypt::Cipher::AES, Crypt::Cipher::Blowfish, ...