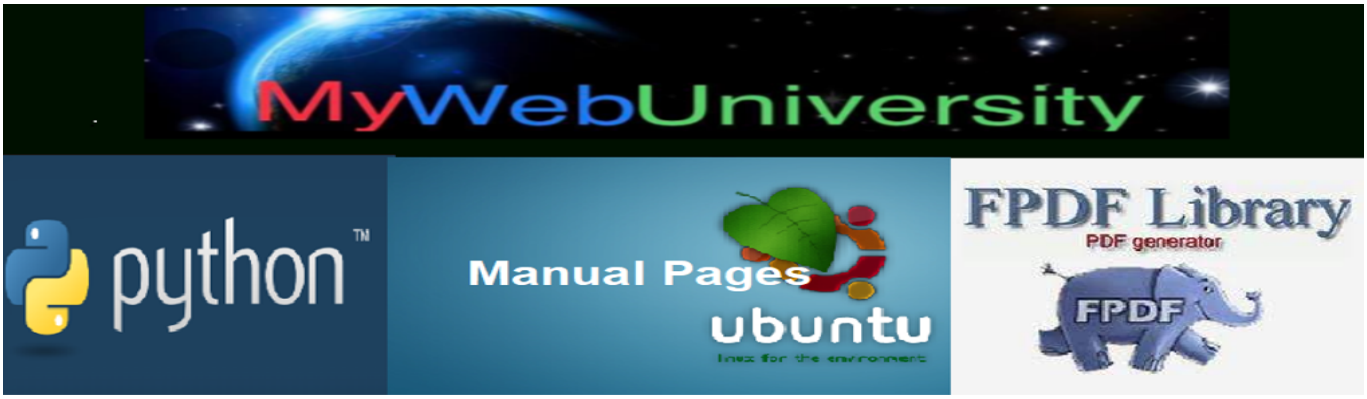

Rocky Enterprise Linux 9.2 Manual Pages on command 'Crypt::AuthEnc::CCM.3pm'

$ man Crypt::AuthEnc::CCM.3pm

Crypt::AuthEnc::CCM(3pm) User Contributed Perl Documentation Crypt::AuthEnc::CCM(3pm)

NAME

Crypt::AuthEnc::CCM - Authenticated encryption in CCM mode

SYNOPSIS

```
### OO interface
use Crypt::AuthEnc::CCM;

# encrypt and authenticate
my $ae = Crypt::AuthEnc::CCM->new("AES", $key, $iv, $adata, $tag_len, $pt_len);
my $ct = $ae->encrypt_add('data1');
$ct .= $ae->encrypt_add('data2');
$ct .= $ae->encrypt_add('data3');
my $tag = $ae->encrypt_done();

# decrypt and verify
my $ae = Crypt::AuthEnc::CCM->new("AES", $key, $iv, $adata, $tag_len, $pt_len);
my $pt = $ae->decrypt_add('ciphertext1');
$pt .= $ae->decrypt_add('ciphertext2');
$pt .= $ae->decrypt_add('ciphertext3');
my $tag = $ae->decrypt_done();
die "decrypt failed" unless $tag eq $expected_tag;
```

```
#or
my $result = $ae->decrypt_done($expected_tag); # 0 or 1

### functional interface
use Crypt::AuthEnc::CCM qw(ccm_encrypt_authenticate ccm_decrypt_verify);

($ciphertext, $tag) = ccm_encrypt_authenticate('AES', $key, $nonce, $adata, $tag_len, $plaintext);
$plaintext = ccm_decrypt_verify('AES', $key, $nonce, $adata, $ciphertext, $tag);
```

DESCRIPTION

CCM is a encrypt+authenticate mode that is centered around using AES (or any 16-byte cipher) as a primitive. Unlike EAX and OCB mode, it is only meant for packet mode where the length of the input is known in advance.

EXPORT

Nothing is exported by default.

You can export selected functions:

```
use Crypt::AuthEnc::CCM qw(ccm_encrypt_authenticate ccm_decrypt_verify);
```

FUNCTIONS

ccm_encrypt_authenticate

```
my ($ciphertext, $tag) = ccm_encrypt_authenticate($cipher, $key, $nonce, $adata, $tag_len, $plaintext);

# $cipher .. 'AES' or name of any other cipher with 16-byte block len
# $key ..... key of proper length (e.g. 128/192/256bits for AES)
# $nonce ... unique nonce/salt (no need to keep it secret)
# $adata ... additional authenticated data
# $tag_len . required length of output tag
```

CCM parameters should follow

<http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38c.pdf>

```
    # tag length:   4, 6, 8, 10, 12, 14, 16 (reasonable minimum is 8)
    # nonce length: 7, 8, 9, 10, 11, 12, 13 (if you are not sure, use 11)
    # BEWARE nonce length determines max. enc/dec data size: max_data_size = 2^(8*(15-nonce_len))

  ccm_decrypt_verify
    my $plaintext = ccm_decrypt_verify($cipher, $key, $nonce, $adata, $ciphertext, $tag);
    # on error returns undef

METHODS
  new
    my $ae = Crypt::AuthEnc::CCM->new($cipher, $key, $nonce, $adata, $tag_len, $pt_len);

    # $cipher .. 'AES' or name of any other cipher with 16-byte block len
    # $key ..... key of proper length (e.g. 128/192/256bits for AES)
    # $nonce ... unique nonce/salt (no need to keep it secret)
    # $adata ... additional authenticated data
    # $tag_len . required length of output tag
    # $pt_len .. expected length of plaintext/ciphertext to encrypt/decrypt

  encrypt_add
    $ciphertext = $ae->encrypt_add($data);       # can be called multiple times

  encrypt_done
    my $tag = $ae->encrypt_done;                 # returns $tag value

  decrypt_add
    $plaintext = $ae->decrypt_add($ciphertext);  # can be called multiple times

  decrypt_done
    my $tag = $ae->decrypt_done;          # returns $tag value
    #or
```

```
my $result = $ae->decrypt_done($tag);  # returns 1 (success) or 0 (failure)
```

clone

```
my $ae_new = $ae->clone;
```

SEE ALSO

- CryptX, Crypt::AuthEnc::EAX, Crypt::AuthEnc::GCM, Crypt::AuthEnc::OCB

- <https://en.wikipedia.org/wiki/CCM_mode>