Rocky Enterprise Linux 9.2 Manual Pages on command 'RSA_meth_dup.3ossl'

$ man RSA_meth_dup.3ossl

RSA_METH_NEW(3ossl) OpenSSL RSA_METH_NEW(3ossl)

NAME

RSA_meth_get0_app_data, RSA_meth_set0_app_data, RSA_meth_new, RSA_meth_free, RSA_meth_dup, RSA_meth_get0_name, RSA_meth_set1_name, RSA_meth_get_flags, RSA_meth_set_flags, RSA_meth_get_pub_enc, RSA_meth_set_pub_enc, RSA_meth_get_pub_dec, RSA_meth_set_pub_dec, RSA_meth_get_priv_enc, RSA_meth_set_priv_enc, RSA_meth_get_priv_dec, RSA_meth_set_priv_dec, RSA_meth_get_mod_exp, RSA_meth_set_mod_exp, RSA_meth_get_bn_mod_exp, RSA_meth_set_bn_mod_exp, RSA_meth_get_init, RSA_meth_set_init, RSA_meth_get_finish, RSA_meth_set_finish, RSA_meth_get_sign, RSA_meth_set_sign, RSA_meth_get_verify, RSA_meth_set_verify, RSA_meth_get_keygen, RSA_meth_set_keygen, RSA_meth_get_multi_prime_keygen, RSA_meth_set_multi_prime_keygen - Routines to build up RSA methods

SYNOPSIS

#include <openssl/rsa.h>

The following functions have been deprecated since OpenSSL 3.0, and can
be hidden entirely by defining OPENSSL_API_COMPAT with a suitable
version value, see openssl_user_macros(7):

```
 RSA_METHOD *RSA_meth_new(const char *name, int flags);
 void RSA_meth_free(RSA_METHOD *meth);

 RSA_METHOD *RSA_meth_dup(const RSA_METHOD *meth);

 const char *RSA_meth_get0_name(const RSA_METHOD *meth);
 int RSA_meth_set1_name(RSA_METHOD *meth, const char *name);

 int RSA_meth_get_flags(const RSA_METHOD *meth);
 int RSA_meth_set_flags(RSA_METHOD *meth, int flags);

 void *RSA_meth_get0_app_data(const RSA_METHOD *meth);
 int RSA_meth_set0_app_data(RSA_METHOD *meth, void *app_data);

 int (*RSA_meth_get_pub_enc(const RSA_METHOD *meth))(int flen, const unsigned char *from,
                                         unsigned char *to, RSA *rsa, int padding);
 int RSA_meth_set_pub_enc(RSA_METHOD *rsa,
                          int (*pub_enc)(int flen, const unsigned char *from,
                                         unsigned char *to, RSA *rsa,
                                         int padding));

 int (*RSA_meth_get_pub_dec(const RSA_METHOD *meth))
     (int flen, const unsigned char *from,
      unsigned char *to, RSA *rsa, int padding);
 int RSA_meth_set_pub_dec(RSA_METHOD *rsa,
                          int (*pub_dec)(int flen, const unsigned char *from,
                                         unsigned char *to, RSA *rsa,
                                         int padding));
```

```
int (*RSA_meth_get_priv_enc(const RSA_METHOD *meth))(int flen, const unsigned char *from,
                                          unsigned char *to, RSA *rsa,
                                          int padding);
int RSA_meth_set_priv_enc(RSA_METHOD *rsa,
                          int (*priv_enc)(int flen, const unsigned char *from,
                                          unsigned char *to, RSA *rsa, int padding));

int (*RSA_meth_get_priv_dec(const RSA_METHOD *meth))(int flen, const unsigned char *from,
                                          unsigned char *to, RSA *rsa,
                                          int padding);
int RSA_meth_set_priv_dec(RSA_METHOD *rsa,
                          int (*priv_dec)(int flen, const unsigned char *from,
                                          unsigned char *to, RSA *rsa, int padding));

/* Can be null */
int (*RSA_meth_get_mod_exp(const RSA_METHOD *meth))(BIGNUM *r0, const BIGNUM *i,
                                         RSA *rsa, BN_CTX *ctx);
int RSA_meth_set_mod_exp(RSA_METHOD *rsa,
                         int (*mod_exp)(BIGNUM *r0, const BIGNUM *i, RSA *rsa,
                                        BN_CTX *ctx));

/* Can be null */
int (*RSA_meth_get_bn_mod_exp(const RSA_METHOD *meth))(BIGNUM *r, const BIGNUM *a,
                                          const BIGNUM *p, const BIGNUM *m,
                                          BN_CTX *ctx, BN_MONT_CTX *m_ctx);
int RSA_meth_set_bn_mod_exp(RSA_METHOD *rsa,
                            int (*bn_mod_exp)(BIGNUM *r, const BIGNUM *a,
                                              const BIGNUM *p, const BIGNUM *m,
                                              BN_CTX *ctx, BN_MONT_CTX *m_ctx));

/* called at new */
int (*RSA_meth_get_init(const RSA_METHOD *meth) (RSA *rsa);
```

```
int RSA_meth_set_init(RSA_METHOD *rsa, int (*init (RSA *rsa));

/* called at free */
int (*RSA_meth_get_finish(const RSA_METHOD *meth))(RSA *rsa);
int RSA_meth_set_finish(RSA_METHOD *rsa, int (*finish)(RSA *rsa));

int (*RSA_meth_get_sign(const RSA_METHOD *meth))(int type, const unsigned char *m,
                                                 unsigned int m_length,
                                                 unsigned char *sigret,
                                                 unsigned int *siglen, const RSA *rsa);
int RSA_meth_set_sign(RSA_METHOD *rsa,
                      int (*sign)(int type, const unsigned char *m,
                                  unsigned int m_length, unsigned char *sigret,
                                  unsigned int *siglen, const RSA *rsa));

int (*RSA_meth_get_verify(const RSA_METHOD *meth))(int dtype, const unsigned char *m,
                                                   unsigned int m_length,
                                                   const unsigned char *sigbuf,
                                                   unsigned int siglen, const RSA *rsa);
int RSA_meth_set_verify(RSA_METHOD *rsa,
                        int (*verify)(int dtype, const unsigned char *m,
                                      unsigned int m_length,
                                      const unsigned char *sigbuf,
                                      unsigned int siglen, const RSA *rsa));

int (*RSA_meth_get_keygen(const RSA_METHOD *meth))(RSA *rsa, int bits, BIGNUM *e,
                                                   BN_GENCB *cb);
int RSA_meth_set_keygen(RSA_METHOD *rsa,
                        int (*keygen)(RSA *rsa, int bits, BIGNUM *e,
                                      BN_GENCB *cb));

int (*RSA_meth_get_multi_prime_keygen(const RSA_METHOD *meth))(RSA *rsa, int bits,
                                                               int primes, BIGNUM *e,
```

BN_GENCB *cb);

int RSA_meth_set_multi_prime_keygen(RSA_METHOD *meth,
int (*keygen) (RSA *rsa, int bits,
int primes, BIGNUM *e,
BN_GENCB *cb));

DESCRIPTION

All of the functions described on this page are deprecated.
Applications should instead use the OSSL_PROVIDER APIs.

The RSA_METHOD type is a structure used for the provision of custom RSA implementations. It provides a set of functions used by OpenSSL for the implementation of the various RSA capabilities.

RSA_meth_new() creates a new RSA_METHOD structure. It should be given a unique name and a set of flags. The name should be a NULL terminated string, which will be duplicated and stored in the RSA_METHOD object. It is the callers responsibility to free the original string. The flags will be used during the construction of a new RSA object based on this RSA_METHOD. Any new RSA object will have those flags set by default.

RSA_meth_dup() creates a duplicate copy of the RSA_METHOD object passed as a parameter. This might be useful for creating a new RSA_METHOD based on an existing one, but with some differences.

RSA_meth_free() destroys an RSA_METHOD structure and frees up any memory associated with it.

RSA_meth_get0_name() will return a pointer to the name of this RSA_METHOD. This is a pointer to the internal name string and so should not be freed by the caller. RSA_meth_set1_name() sets the name of the RSA_METHOD to name. The string is duplicated and the copy is stored in

the RSA_METHOD structure, so the caller remains responsible for freeing the memory associated with the name.

RSA_meth_get_flags() returns the current value of the flags associated with this RSA_METHOD. RSA_meth_set_flags() provides the ability to set these flags.

The functions RSA_meth_get0_app_data() and RSA_meth_set0_app_data() provide the ability to associate implementation specific data with the RSA_METHOD. It is the application's responsibility to free this data before the RSA_METHOD is freed via a call to RSA_meth_free().

RSA_meth_get_sign() and RSA_meth_set_sign() get and set the function used for creating an RSA signature respectively. This function will be called in response to the application calling RSA_sign(). The parameters for the function have the same meaning as for RSA_sign().

RSA_meth_get_verify() and RSA_meth_set_verify() get and set the function used for verifying an RSA signature respectively. This function will be called in response to the application calling RSA_verify(). The parameters for the function have the same meaning as for RSA_verify().

RSA_meth_get_mod_exp() and RSA_meth_set_mod_exp() get and set the function used for CRT computations.

RSA_meth_get_bn_mod_exp() and RSA_meth_set_bn_mod_exp() get and set the function used for CRT computations, specifically the following value:

r = a ^ p mod m

Both the mod_exp() and bn_mod_exp() functions are called by the default OpenSSL method during encryption, decryption, signing and verification.

RSA_meth_get_init() and RSA_meth_set_init() get and set the function used for creating a new RSA instance respectively. This function will be called in response to the application calling RSA_new() (if the current default RSA_METHOD is this one) or RSA_new_method(). The RSA_new() and RSA_new_method() functions will allocate the memory for the new RSA object, and a pointer to this newly allocated structure will be passed as a parameter to the function. This function may be NULL.

RSA_meth_get_finish() and RSA_meth_set_finish() get and set the function used for destroying an instance of an RSA object respectively. This function will be called in response to the application calling RSA_free(). A pointer to the RSA to be destroyed is passed as a parameter. The destroy function should be used for RSA implementation specific clean up. The memory for the RSA itself should not be freed by this function. This function may be NULL.

RSA_meth_get_keygen() and RSA_meth_set_keygen() get and set the function used for generating a new RSA key pair respectively. This function will be called in response to the application calling RSA_generate_key_ex(). The parameter for the function has the same meaning as for RSA_generate_key_ex().

RSA_meth_get_multi_prime_keygen() and RSA_meth_set_multi_prime_keygen() get and set the function used for generating a new multi-prime RSA key pair respectively. This function will be called in response to the application calling RSA_generate_multi_prime_key(). The parameter for the function has the same meaning as for RSA_generate_multi_prime_key().

RSA_meth_get_pub_enc(), RSA_meth_set_pub_enc(), RSA_meth_get_pub_dec(), RSA_meth_set_pub_dec(), RSA_meth_get_priv_enc(),

RSA_meth_set_priv_enc(), RSA_meth_get_priv_dec(),
RSA_meth_set_priv_dec() get and set the functions used for public and
private key encryption and decryption. These functions will be called
in response to the application calling RSA_public_encrypt(),
RSA_private_decrypt(), RSA_private_encrypt() and RSA_public_decrypt()
and take the same parameters as those.

RETURN VALUES

RSA_meth_new() and RSA_meth_dup() return the newly allocated RSA_METHOD
object or NULL on failure.

RSA_meth_get0_name() and RSA_meth_get_flags() return the name and flags
associated with the RSA_METHOD respectively.

All other RSA_meth_get_*() functions return the appropriate function
pointer that has been set in the RSA_METHOD, or NULL if no such pointer
has yet been set.

RSA_meth_set1_name and all RSA_meth_set_*() functions return 1 on
success or 0 on failure.

SEE ALSO

RSA_new(3), RSA_generate_key_ex(3), RSA_sign(3), RSA_set_method(3),
RSA_size(3), RSA_get0_key(3), RSA_generate_multi_prime_key(3)

HISTORY

All of these functions were deprecated in OpenSSL 3.0.

RSA_meth_get_multi_prime_keygen() and RSA_meth_set_multi_prime_keygen()
were added in OpenSSL 1.1.1.

Other functions described here were added in OpenSSL 1.1.0.

COPYRIGHT