



Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'BN_BLINDING_convert.3ossl' command

\$ man BN_BLINDING_convert.3ossl

BN_BLINDING_NEW(3ossl) OpenSSL BN_BLINDING_NEW(3ossl)

NAME

BN_BLINDING_new, BN_BLINDING_free, BN_BLINDING_update,
BN_BLINDING_convert, BN_BLINDING_invert, BN_BLINDING_convert_ex,
BN_BLINDING_invert_ex, BN_BLINDING_is_current_thread,
BN_BLINDING_set_current_thread, BN_BLINDING_lock, BN_BLINDING_unlock,
BN_BLINDING_get_flags, BN_BLINDING_set_flags, BN_BLINDING_create_param
- blinding related BIGNUM functions

SYNOPSIS

```
#include <openssl/bn.h>

BN_BLINDING *BN_BLINDING_new(const BIGNUM *A, const BIGNUM *Ai,
                             BIGNUM *mod);

void BN_BLINDING_free(BN_BLINDING *b);

int BN_BLINDING_update(BN_BLINDING *b, BN_CTX *ctx);

int BN_BLINDING_convert(BIGNUM *n, BN_BLINDING *b, BN_CTX *ctx);

int BN_BLINDING_invert(BIGNUM *n, BN_BLINDING *b, BN_CTX *ctx);

int BN_BLINDING_convert_ex(BIGNUM *n, BIGNUM *r, BN_BLINDING *b,
                           BN_CTX *ctx);

int BN_BLINDING_invert_ex(BIGNUM *n, const BIGNUM *r, BN_BLINDING *b,
                          BN_CTX *ctx);
```

```

int BN_BLINDING_is_current_thread(BN_BLINDING *b);
void BN_BLINDING_set_current_thread(BN_BLINDING *b);
int BN_BLINDING_lock(BN_BLINDING *b);
int BN_BLINDING_unlock(BN_BLINDING *b);
unsigned long BN_BLINDING_get_flags(const BN_BLINDING *b);
void BN_BLINDING_set_flags(BN_BLINDING *b, unsigned long flags);
BN_BLINDING *BN_BLINDING_create_param(BN_BLINDING *b,
                                     const BIGNUM *e, BIGNUM *m, BN_CTX *ctx,
                                     int (*bn_mod_exp)(BIGNUM *r,
                                                         const BIGNUM *a,
                                                         const BIGNUM *p,
                                                         const BIGNUM *m,
                                                         BN_CTX *ctx,
                                                         BN_MONT_CTX *m_ctx),
                                     BN_MONT_CTX *m_ctx);

```

DESCRIPTION

BN_BLINDING_new() allocates a new BN_BLINDING structure and copies the A and Ai values into the newly created BN_BLINDING object.

BN_BLINDING_free() frees the BN_BLINDING structure. If b is NULL, nothing is done.

BN_BLINDING_update() updates the BN_BLINDING parameters by squaring the A and Ai or, after specific number of uses and if the necessary parameters are set, by re-creating the blinding parameters.

BN_BLINDING_convert_ex() multiplies n with the blinding factor A. If r is not NULL a copy the inverse blinding factor Ai will be returned in r (this is useful if a RSA object is shared among several threads).

BN_BLINDING_invert_ex() multiplies n with the inverse blinding factor Ai. If r is not NULL it will be used as the inverse blinding.

BN_BLINDING_convert() and BN_BLINDING_invert() are wrapper functions for BN_BLINDING_convert_ex() and BN_BLINDING_invert_ex() with r set to NULL.

BN_BLINDING_is_current_thread() returns whether the BN_BLINDING structure is owned by the current thread. This is to help users provide proper locking if needed for multi-threaded use.

BN_BLINDING_set_current_thread() sets the current thread as the owner of the BN_BLINDING structure.

BN_BLINDING_lock() locks the BN_BLINDING structure.

BN_BLINDING_unlock() unlocks the BN_BLINDING structure.

BN_BLINDING_get_flags() returns the BN_BLINDING flags. Currently there are two supported flags: BN_BLINDING_NO_UPDATE and BN_BLINDING_NO_RECREATE. BN_BLINDING_NO_UPDATE inhibits the automatic update of the BN_BLINDING parameters after each use and BN_BLINDING_NO_RECREATE inhibits the automatic re-creation of the BN_BLINDING parameters after a fixed number of uses (currently 32). In newly allocated BN_BLINDING objects no flags are set.

BN_BLINDING_set_flags() sets the BN_BLINDING parameters flags.

BN_BLINDING_create_param() creates new BN_BLINDING parameters using the exponent e and the modulus m. bn_mod_exp and m_ctx can be used to pass special functions for exponentiation (normally BN_mod_exp_mont() and BN_MONT_CTX).

RETURN VALUES

BN_BLINDING_new() returns the newly allocated BN_BLINDING structure or NULL in case of an error.

BN_BLINDING_update(), BN_BLINDING_convert(), BN_BLINDING_invert(), BN_BLINDING_convert_ex() and BN_BLINDING_invert_ex() return 1 on success and 0 if an error occurred.

BN_BLINDING_is_current_thread() returns 1 if the current thread owns the BN_BLINDING object, 0 otherwise.

BN_BLINDING_set_current_thread() doesn't return anything.

BN_BLINDING_lock(), BN_BLINDING_unlock() return 1 if the operation succeeded or 0 on error.

BN_BLINDING_get_flags() returns the currently set BN_BLINDING flags (a unsigned long value).

BN_BLINDING_create_param() returns the newly created BN_BLINDING parameters or NULL on error.

HISTORY

BN_BLINDING_thread_id() was first introduced in OpenSSL 1.0.0, and it deprecates BN_BLINDING_set_thread_id() and BN_BLINDING_get_thread_id().

COPYRIGHT

Copyright 2005-2021 The OpenSSL Project Authors. 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>.