



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

***Rocky Enterprise Linux 9.2 Manual Pages on command 'BN\_mod\_exp\_mont\_consttime\_x2.3ossl'***

***\$ man BN\_mod\_exp\_mont\_consttime\_x2.3ossl***

BN\_MOD\_EXP\_MONT(3ossl)          OpenSSL          BN\_MOD\_EXP\_MONT(3ossl)

NAME

BN\_mod\_exp\_mont, BN\_mod\_exp\_mont\_consttime,  
BN\_mod\_exp\_mont\_consttime\_x2 - Montgomery exponentiation

SYNOPSIS

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

```
int BN_mod_exp_mont(BIGNUM *rr, const BIGNUM *a, const BIGNUM *p,  
                  const BIGNUM *m, BN_CTX *ctx, BN_MONT_CTX *in_mont);
```

```
int BN_mod_exp_mont_consttime(BIGNUM *rr, const BIGNUM *a, const BIGNUM *p,  
                              const BIGNUM *m, BN_CTX *ctx,  
                              BN_MONT_CTX *in_mont);
```

```
int BN_mod_exp_mont_consttime_x2(BIGNUM *rr1, const BIGNUM *a1,  
                                  const BIGNUM *p1, const BIGNUM *m1,
```

```
BN_MONT_CTX *in_mont1, BIGNUM *rr2,  
const BIGNUM *a2, const BIGNUM *p2,  
const BIGNUM *m2, BN_MONT_CTX *in_mont2,  
BN_CTX *ctx);
```

## DESCRIPTION

`BN_mod_exp_mont()` computes  $a$  to the  $p$ -th power modulo  $m$  (" $rr=a^p \% m$ ") using Montgomery multiplication. `in_mont` is a Montgomery context and can be `NULL`. In the case `in_mont` is `NULL`, it will be initialized within the function, so you can save time on initialization if you provide it in advance.

`BN_mod_exp_mont_consttime()` computes  $a$  to the  $p$ -th power modulo  $m$  (" $rr=a^p \% m$ ") using Montgomery multiplication. It is a variant of `BN_mod_exp_mont(3)` that uses fixed windows and the special precomputation memory layout to limit data-dependency to a minimum to protect secret exponents. It is called automatically when `BN_mod_exp_mont(3)` is called with parameters  $a$ ,  $p$ ,  $m$ , any of which have `BN_FLG_CONSTTIME` flag.

`BN_mod_exp_mont_consttime_x2()` computes two independent exponentiations  $a_1$  to the  $p_1$ -th power modulo  $m_1$  (" $rr1=a_1^{p_1} \% m_1$ ") and  $a_2$  to the  $p_2$ -th power modulo  $m_2$  (" $rr2=a_2^{p_2} \% m_2$ ") using Montgomery multiplication. For some fixed and equal modulus sizes  $m_1$  and  $m_2$  it uses optimizations that allow to speedup two exponentiations. In all other cases the function reduces to two calls of `BN_mod_exp_mont_consttime(3)`.

## RETURN VALUES

For all functions 1 is returned for success, 0 on error. The error codes can be obtained by `ERR_get_error(3)`.

## SEE ALSO

`ERR_get_error(3)`, `BN_mod_exp_mont(3)`

## COPYRIGHT

Copyright 2000-2021 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>.

3.0.7                    2023-07-13            BN\_MOD\_EXP\_MONT(3ossl)