

**NAME**

BN\_bn2bin, BN\_bin2bn, BN\_bn2hex, BN\_bn2dec, BN\_hex2bn, BN\_dec2bn, BN\_print, BN\_print\_fp, BN\_bn2mpi, BN\_mpi2bn - format conversions

**SYNOPSIS**

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

int BN_bn2bin(const BIGNUM *a, unsigned char *to);
BIGNUM *BN_bin2bn(const unsigned char *s, int len, BIGNUM *ret);

char *BN_bn2hex(const BIGNUM *a);
char *BN_bn2dec(const BIGNUM *a);
int BN_hex2bn(BIGNUM **a, const char *str);
int BN_dec2bn(BIGNUM **a, const char *str);

int BN_print(BIO *fp, const BIGNUM *a);
int BN_print_fp(FILE *fp, const BIGNUM *a);

int BN_bn2mpi(const BIGNUM *a, unsigned char *to);
BIGNUM *BN_mpi2bn(unsigned char *s, int len, BIGNUM *ret);
```

**DESCRIPTION**

*BN\_bn2bin()* converts the absolute value of **a** into big-endian form and stores it at **to**. **to** must point to `BN_num_bytes(a)` bytes of memory.

*BN\_bin2bn()* converts the positive integer in big-endian form of length **len** at **s** into a **BIGNUM** and places it in **ret**. If **ret** is NULL, a new **BIGNUM** is created.

*BN\_bn2hex()* and *BN\_bn2dec()* return printable strings containing the hexadecimal and decimal encoding of **a** respectively. For negative numbers, the string is prefaced with a leading '-'. The string must be freed later using *OPENSSL\_free()*.

*BN\_hex2bn()* converts the string **str** containing a hexadecimal number to a **BIGNUM** and stores it in **\*\*bn**. If **\*bn** is NULL, a new **BIGNUM** is created. If **bn** is NULL, it only computes the number's length in hexadecimal digits. If the string starts with '-', the number is negative. *BN\_dec2bn()* is the same using the decimal system.

*BN\_print()* and *BN\_print\_fp()* write the hexadecimal encoding of **a**, with a leading '-' for negative numbers, to the **BIO** or **FILE fp**.

*BN\_bn2mpi()* and *BN\_mpi2bn()* convert **BIGNUM**s from and to a format that consists of the number's length in bytes represented as a 4-byte big-endian number, and the number itself in big-endian format, where the most significant bit signals a negative number (the representation of numbers with the MSB set is prefixed with null byte).

*BN\_bn2mpi()* stores the representation of **a** at **to**, where **to** must be large enough to hold the result. The size can be determined by calling `BN_bn2mpi(a, NULL)`.

*BN\_mpi2bn()* converts the **len** bytes long representation at **s** to a **BIGNUM** and stores it at **ret**, or in a newly allocated **BIGNUM** if **ret** is NULL.

**RETURN VALUES**

*BN\_bn2bin()* returns the length of the big-endian number placed at **to**. *BN\_bin2bn()* returns the **BIGNUM**, NULL on error.

*BN\_bn2hex()* and *BN\_bn2dec()* return a null-terminated string, or NULL on error. *BN\_hex2bn()* and *BN\_dec2bn()* return the number's length in hexadecimal or decimal digits, and 0 on error.

*BN\_print\_fp()* and *BN\_print()* return 1 on success, 0 on write errors.

*BN\_bn2mpi()* returns the length of the representation. *BN\_mpi2bn()* returns the **BIGNUM**, and NULL on error.

The error codes can be obtained by [ERR\\_get\\_error\(3\)](#).

**SEE ALSO**

[bn\(3\)](#), [ERR\\_get\\_error\(3\)](#), [BN\\_zero\(3\)](#), [ASN1\\_INTEGER\\_to\\_BN\(3\)](#), [BN\\_num\\_bytes\(3\)](#)

**HISTORY**

[BN\\_bn2bin\(\)](#), [BN\\_bin2bn\(\)](#), [BN\\_print\\_fp\(\)](#) and [BN\\_print\(\)](#) are available in all versions of SSLey and OpenSSL.

[BN\\_bn2hex\(\)](#), [BN\\_bn2dec\(\)](#), [BN\\_hex2bn\(\)](#), [BN\\_dec2bn\(\)](#), [BN\\_bn2mpi\(\)](#) and [BN\\_mpi2bn\(\)](#) were added in SSLey 0.9.0.