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

d Hat Enterprise Linux Release 9.2 Manual Pages on 'OSSL_PARAM_BLD_push_octet_string.3ossl' comma

$ man OSSL_PARAM_BLD_push_octet_string.3ossl

OSSL_PARAM_BLD(3ossl) OpenSSL OSSL_PARAM_BLD(3ossl)

NAME

OSSL_PARAM_BLD, OSSL_PARAM_BLD_new, OSSL_PARAM_BLD_to_param,
OSSL_PARAM_BLD_free, OSSL_PARAM_BLD_push_int, OSSL_PARAM_BLD_push_uint,
OSSL_PARAM_BLD_push_long, OSSL_PARAM_BLD_push_ulong,
OSSL_PARAM_BLD_push_int32, OSSL_PARAM_BLD_push_uint32,
OSSL_PARAM_BLD_push_int64, OSSL_PARAM_BLD_push_uint64,
OSSL_PARAM_BLD_push_size_t, OSSL_PARAM_BLD_push_time_t,
OSSL_PARAM_BLD_push_double, OSSL_PARAM_BLD_push_BN,
OSSL_PARAM_BLD_push_BN_pad, OSSL_PARAM_BLD_push_utf8_string,
OSSL_PARAM_BLD_push_utf8_ptr, OSSL_PARAM_BLD_push_octet_string,
OSSL_PARAM_BLD_push_octet_ptr - functions to assist in the creation of
OSSL_PARAM arrays

SYNOPSIS

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

typedef struct OSSL_PARAM_BLD;

OSSL_PARAM_BLD *OSSL_PARAM_BLD_new(void);
OSSL_PARAM *OSSL_PARAM_BLD_to_param(OSSL_PARAM_BLD *bld);
void OSSL_PARAM_BLD_free(OSSL_PARAM_BLD *bld);
```

```
int OSSL_PARAM_BLD_push_TYPE(OSSL_PARAM_BLD *bld, const char *key, TYPE val);

int OSSL_PARAM_BLD_push_BN(OSSL_PARAM_BLD *bld, const char *key,
                           const BIGNUM *bn);
int OSSL_PARAM_BLD_push_BN_pad(OSSL_PARAM_BLD *bld, const char *key,
                               const BIGNUM *bn, size_t sz);

int OSSL_PARAM_BLD_push_utf8_string(OSSL_PARAM_BLD *bld, const char *key,
                                    const char *buf, size_t bsize);
int OSSL_PARAM_BLD_push_utf8_ptr(OSSL_PARAM_BLD *bld, const char *key,
                                 char *buf, size_t bsize);
int OSSL_PARAM_BLD_push_octet_string(OSSL_PARAM_BLD *bld, const char *key,
                                     const void *buf, size_t bsize);
int OSSL_PARAM_BLD_push_octet_ptr(OSSL_PARAM_BLD *bld, const char *key,
                                  void *buf, size_t bsize);
```

DESCRIPTION

A collection of utility functions that simplify the creation of OSSL_PARAM arrays. The TYPE names are as per OSSL_PARAM_int(3).

OSSL_PARAM_BLD_new() allocates and initialises a new OSSL_PARAM_BLD structure so that values can be added. Any existing values are cleared.

OSSL_PARAM_BLD_free() deallocates the memory allocates by OSSL_PARAM_BLD_new().

OSSL_PARAM_BLD_to_param() converts a built up OSSL_PARAM_BLD structure bld into an allocated OSSL_PARAM array. The OSSL_PARAM array and all associated storage must be freed by calling OSSL_PARAM_free() with the functions return value. OSSL_PARAM_BLD_free() can safely be called any time after this function is.

OSSL_PARAM_BLD_push_TYPE() are a series of functions which will create OSSL_PARAM objects of the specified size and correct type for the val argument. val is stored by value and an expression or auto variable can be used.

OSSL_PARAM_BLD_push_BN() is a function that will create an OSSL_PARAM object that holds the specified BIGNUM bn. If bn is marked as being securely allocated, its OSSL_PARAM representation will also be securely allocated. The bn argument is stored by reference and the underlying BIGNUM object must exist until after OSSL_PARAM_BLD_to_param() has been called.

OSSL_PARAM_BLD_push_BN_pad() is a function that will create an OSSL_PARAM object that holds the specified BIGNUM bn. The object will be padded to occupy exactly sz bytes, if insufficient space is specified an error results. If bn is marked as being securely allocated, its OSSL_PARAM representation will also be securely allocated. The bn argument is stored by reference and the underlying BIGNUM object must exist until after OSSL_PARAM_BLD_to_param() has been called.

OSSL_PARAM_BLD_push_utf8_string() is a function that will create an OSSL_PARAM object that references the UTF8 string specified by buf. The length of the string bsize should not include the terminating NUL byte. If it is zero then it will be calculated. The string that buf points to is stored by reference and must remain in scope until after OSSL_PARAM_BLD_to_param() has been called.

OSSL_PARAM_BLD_push_octet_string() is a function that will create an OSSL_PARAM object that references the octet string specified by buf and <bsize>. The memory that buf points to is stored by reference and must remain in scope until after OSSL_PARAM_BLD_to_param() has been called.

OSSL_PARAM_BLD_push_utf8_ptr() is a function that will create an OSSL_PARAM object that references the UTF8 string specified by buf. The length of the string bsize should not include the terminating NUL byte. If it is zero then it will be calculated. The string buf points to is stored by reference and must remain in scope until the OSSL_PARAM array is freed.

OSSL_PARAM_BLD_push_octet_ptr() is a function that will create an OSSL_PARAM object that references the octet string specified by buf. The memory buf points to is stored by reference and must remain in scope until the OSSL_PARAM array is freed.

RETURN VALUES

OSSL_PARAM_BLD_new() returns the allocated OSSL_PARAM_BLD structure, or NULL on error.

OSSL_PARAM_BLD_to_param() returns the allocated OSSL_PARAM array, or NULL on error.

All of the OSSL_PARAM_BLD_push_TYPE functions return 1 on success and 0 on error.

NOTES

OSSL_PARAM_BLD_push_BN() and OSSL_PARAM_BLD_push_BN_pad() currently only support nonnegative BIGNUMs. They return an error on negative BIGNUMs.

EXAMPLES

Both examples creating an OSSL_PARAM array that contains an RSA key. For both, the predefined key variables are:

```
BIGNUM *n;          /* modulus */
```

```
    unsigned int e;      /* public exponent */
    BIGNUM *d;           /* private exponent */
    BIGNUM *p, *q;       /* first two prime factors */
    BIGNUM *dmp1, *dmq1; /* first two CRT exponents */
    BIGNUM *iqmp;        /* first CRT coefficient */
```

Example 1

This example shows how to create an OSSL_PARAM array that contains an RSA private key.

```
    OSSL_PARAM_BLD *bld = OSSL_PARAM_BLD_new();
    OSSL_PARAM *params = NULL;

    if (bld == NULL
        || !OSSL_PARAM_BLD_push_BN(bld, "n", n)
        || !OSSL_PARAM_BLD_push_uint(bld, "e", e)
        || !OSSL_PARAM_BLD_push_BN(bld, "d", d)
        || !OSSL_PARAM_BLD_push_BN(bld, "rsa-factor1", p)
        || !OSSL_PARAM_BLD_push_BN(bld, "rsa-factor2", q)
        || !OSSL_PARAM_BLD_push_BN(bld, "rsa-exponent1", dmp1)
        || !OSSL_PARAM_BLD_push_BN(bld, "rsa-exponent2", dmq1)
        || !OSSL_PARAM_BLD_push_BN(bld, "rsa-coefficient1", iqmp)
        || (params = OSSL_PARAM_BLD_to_param(bld)) == NULL)
        goto err;
    OSSL_PARAM_BLD_free(bld);
    /* Use params */
    ...
    OSSL_PARAM_free(params);
```

Example 2

This example shows how to create an OSSL_PARAM array that contains an RSA public key.

```
OSSL_PARAM_BLD *bld = OSSL_PARAM_BLD_new();
OSSL_PARAM *params = NULL;

if (nld == NULL
    || !OSSL_PARAM_BLD_push_BN(bld, "n", n)
    || !OSSL_PARAM_BLD_push_uint(bld, "e", e)
    || (params = OSSL_PARAM_BLD_to_param(bld)) == NULL)
    goto err;
OSSL_PARAM_BLD_free(bld);
/* Use params */
...
OSSL_PARAM_free(params);
```

SEE ALSO

OSSL_PARAM_int(3), OSSL_PARAM(3), OSSL_PARAM_free(3)

HISTORY

The functions described here were all added in OpenSSL 3.0.

COPYRIGHT