

**NAME**

DSA\_generate\_parameters\_ex, DSA\_generate\_parameters - generate DSA parameters

**SYNOPSIS**

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

```
int DSA_generate_parameters_ex(DSA *dsa, int bits,
    const unsigned char *seed, int seed_len,
    int *counter_ret, unsigned long *h_ret, BN_GENCB *cb);
```

Deprecated:

```
DSA *DSA_generate_parameters(int bits, unsigned char *seed,
    int seed_len, int *counter_ret, unsigned long *h_ret,
    void (*callback)(int, int, void *), void *cb_arg);
```

**DESCRIPTION**

*DSA\_generate\_parameters\_ex()* generates primes *p* and *q* and a generator *g* for use in the DSA and stores the result in **dsa**.

**bits** is the length of the prime to be generated; the DSS allows a maximum of 1024 bits.

If **seed** is **NULL** or **seed\_len** < 20, the primes will be generated at random. Otherwise, the seed is used to generate them. If the given seed does not yield a prime *q*, a new random seed is chosen and placed at **seed**.

*DSA\_generate\_parameters\_ex()* places the iteration count in **\*counter\_ret** and a counter used for finding a generator in **\*h\_ret**, unless these are **NULL**.

A callback function may be used to provide feedback about the progress of the key generation. If **cb** is not **NULL**, it will be called as shown below. For information on the **BN\_GENCB** structure and the **BN\_GENCB\_call** function discussed below, refer to [BN\\_generate\\_prime\(3\)](#).

- When a candidate for *q* is generated, **BN\_GENCB\_call(cb, 0, m++)** is called (*m* is 0 for the first candidate).
- When a candidate for *q* has passed a test by trial division, **BN\_GENCB\_call(cb, 1, -1)** is called. While a candidate for *q* is tested by Miller-Rabin primality tests, **BN\_GENCB\_call(cb, 1, i)** is called in the outer loop (once for each witness that confirms that the candidate may be prime); *i* is the loop counter (starting at 0).
- When a prime *q* has been found, **BN\_GENCB\_call(cb, 2, 0)** and **BN\_GENCB\_call(cb, 3, 0)** are called.
- Before a candidate for *p* (other than the first) is generated and tested, **BN\_GENCB\_call(cb, 0, counter)** is called.
- When a candidate for *p* has passed the test by trial division, **BN\_GENCB\_call(cb, 1, -1)** is called. While it is tested by the Miller-Rabin primality test, **BN\_GENCB\_call(cb, 1, i)** is called in the outer loop (once for each witness that confirms that the candidate may be prime). *i* is the loop counter (starting at 0).
- When *p* has been found, **BN\_GENCB\_call(cb, 2, 1)** is called.
- When the generator has been found, **BN\_GENCB\_call(cb, 3, 1)** is called.

*DSA\_generate\_parameters()* (deprecated) works in much the same way as for *DSA\_generate\_parameters\_ex*, except that no **dsa** parameter is passed and instead a newly allocated **DSA** structure is returned. Additionally “old style” callbacks are used instead of the newer **BN\_GENCB** based approach. Refer to [BN\\_generate\\_prime\(3\)](#) for further information.

**RETURN VALUE**

*DSA\_generate\_parameters\_ex()* returns a 1 on success, or 0 otherwise.

*DSA\_generate\_parameters()* returns a pointer to the DSA structure, or **NULL** if the parameter

generation fails.

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

## BUGS

Seed lengths > 20 are not supported.

## SEE ALSO

[dsa\(3\)](#), [ERR\\_get\\_error\(3\)](#), [rand\(3\)](#), [DSA\\_free\(3\)](#), [BN\\_generate\\_prime\(3\)](#)

## HISTORY

[DSA\\_generate\\_parameters\(\)](#) appeared in SSLeay 0.8. The **cb\_arg** argument was added in SSLeay 0.9.0. In versions up to OpenSSL 0.9.4, **callback(1, ...)** was called in the inner loop of the Miller-Rabin test whenever it reached the squaring step (the parameters to **callback** did not reveal how many witnesses had been tested); since OpenSSL 0.9.5, **callback(1, ...)** is called as in [BN\\_is\\_prime\(3\)](#), i.e. once for each witness.