NAME

genrsa - generate an RSA private key

SYNOPSIS

openssl genrsa [**-out filename**] [**-passout arg**] [**-aes128**] [**-aes128**] [**-aes192**] [**-aes256**] [**-camellia128**] [**-camellia192**] [**-camellia256**] [**-aes192**] [**-aes256**] [**-camellia128**] [**-camellia192**] [**-camellia256**] [**-des**] [**-des3**] [**-idea**] [**-f4**] [**-3**] [**-rand file(s)**] [**-engine id**] [**numbits**]

DESCRIPTION

The **genrsa** command generates an RSA private key.

OPTIONS

-out filename

the output filename. If this argument is not specified then standard output is used.

-passout arg

the output file password source. For more information about the format of **arg** see the **PASS PHRASE ARGUMENTS** section in *openssl(1)*.

-aes128|-aes192|-aes256|-camellia128|-camellia192|-camellia256|-des|-des3|-idea

These options encrypt the private key with specified cipher before outputting it. If none of these options is specified no encryption is used. If encryption is used a pass phrase is prompted for if it is not supplied via the **-passout** argument.

-F4|-3

the public exponent to use, either 65537 or 3. The default is 65537.

-rand file(s)

a file or files containing random data used to seed the random number generator, or an EGD socket (see *RAND_egd(3)*). Multiple files can be specified separated by a OS-dependent character. The separator is**;** for MS-Windo ws, **,** for OpenVMS, and **:** for all others.

-engine id

specifying an engine (by its unique **id** string) will cause **genrsa** to attempt to obtain a functional reference to the specified engine, thus initialising it if needed. The engine will then be set as the default for all available algorithms.

numbits

the size of the private key to generate in bits. This must be the last option specified. The default is 512.

NOTES

RSA private key generation essentially involves the generation of two prime numbers. When generating a private key various symbols will be output to indicate the progress of the generation. A **.** represents each number which has passed an initial sieve test, **+** means a number has passed a single round of the Miller-Rabin primality test. A newline means that the number has passed all the prime tests (the actual number depends on the key size).

Because key generation is a random process the time taken to generate a key may vary somewhat.

BUGS

A quirk of the prime generation algorithm is that it cannot generate small primes. Therefore the number of bits should not be less that 64. For typical private keys this will not matter because for security reasons they will be much larger (typically 1024 bits).

SEE ALSO

gendsa(1)