NAME

RAND_egd, RAND_egd_bytes, RAND_query_egd_bytes - query entropy gathering daemon

SYNOPSIS

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

int RAND_egd(const char *path);
int RAND_egd_bytes(const char *path, int bytes);

int RAND_query_egd_bytes(const char *path, unsigned char *buf, int bytes);
```

DESCRIPTION

RAND_egd() queries the entropy gathering daemon EGD on socket **path**. It queries 255 bytes and uses *RAND_add(3)* to seed the OpenSSL built-in PRNG. RAND_egd(path) is a wrapper for RAND_egd_bytes(path, 255);

RAND_egd_bytes() queries the entropy gathering daemon EGD on socket **path**. It queries**bytes** bytes and uses *RAND_add(3)* to seed the OpenSSL built-in PRNG. This function is more flexible than *RAND_egd()*. When only one secret key must be generated, it is not necessary to request the full amount 255 bytes from the EGD socket. This can be advantageous, since the amount of entropy that can be retrieved from EGD over time is limited.

RAND_query_egd_bytes() performs the actual query of the EGD daemon on socket **path**. If **buf** is given, **bytes** bytes are queried and written into **buf**. If **buf** is NULL, **bytes** bytes are queried and used to seed the OpenSSL built-in PRNG using *RAND_add(3)* .

NOTES

On systems without /dev/*random devices providing entropy from the kernel, the EGD entropy gathering daemon can be used to collect entropy. It provides a socket interface through which entropy can be gathered in chunks up to 255 bytes. Several chunks can be queried during one connection.

EGD is available from http://www.lothar.com/tech/crypto/ (`perl Makefile.PL; make; make install` to install). It is run as **egd** *path*, where *path* is an absolute path designating a socket. When *RAND_egd()* is called with that path as an argument, it tries to read random bytes that EGD has collected. *RAND_egd()* retrieves entropy from the daemon using the daemon's ''non-blocking read'' command which shall be answered immediately by the daemon without waiting for additional entropy to be collected. The write and read socket operations in the communication are blocking.

Alternatively, the EGD-interface compatible daemon PRNGD can be used. It is available from http://prngd.sourceforge.net/ PRNGD does employ an internal PRNG itself and can therefore never run out of entropy.

OpenSSL automatically queries EGD when entropy is requested via *RAND_bytes()* or the status is checked via *RAND_status()* for the first time, if the socket is located at /var/run/egd-pool, /dev/egd-pool or /etc/egd-pool.

RETURN VALUE

RAND_egd() and *RAND_egd_bytes()* return the number of bytes read from the daemon on success, and -1 if the connection failed or the daemon did not return enough data to fully seed the PRNG.

RAND_query_egd_bytes() returns the number of bytes read from the daemon on success, and -1 if the connection failed. The PRNG state is not considered.

SEE ALSO

rand(3) , *RAND_add(3)* , *RAND_cleanup(3)*

COPYRIGHT