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

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'EVP_RAND-TEST-RAND.7ossl' command

$ man EVP_RAND-TEST-RAND.7ossl

EVP_RAND-TEST-RAND(7ossl) OpenSSL EVP_RAND-TEST-RAND(7ossl)

NAME

EVP_RAND-TEST-RAND - The test EVP_RAND implementation

DESCRIPTION

Support for a test generator through the EVP_RAND API. This generator is for test purposes only, it does not generate random numbers.

Identity

"TEST-RAND" is the name for this implementation; it can be used with the EVP_RAND_fetch() function.

Supported parameters

The supported parameters are:

"state" (OSSL_RAND_PARAM_STATE) <integer>

These parameter works as described in "PARAMETERS" in EVP_RAND(3).

"strength" (OSSL_RAND_PARAM_STRENGTH) <unsigned integer>

"reseed_requests" (OSSL_DRBG_PARAM_RESEED_REQUESTS) <unsigned integer>

"reseed_time_interval" (OSSL_DRBG_PARAM_RESEED_TIME_INTERVAL) <integer>

"max_request" (OSSL_DRBG_PARAM_RESEED_REQUESTS) <unsigned integer>

"min_entropylen" (OSSL_DRBG_PARAM_MIN_ENTROPYLEN) <unsigned integer>

"max_entropylen" (OSSL_DRBG_PARAM_MAX_ENTROPYLEN) <unsigned integer>

"min_noncelen" (OSSL_DRBG_PARAM_MIN_NONCELEN) <unsigned integer>

"max_noncelen" (OSSL_DRBG_PARAM_MAX_NONCELEN) <unsigned integer>

"max_perslen" (OSSL_DRBG_PARAM_MAX_PERSLEN) <unsigned integer>

"max_adinlen" (OSSL_DRBG_PARAM_MAX_ADINLEN) <unsigned integer>

"reseed_counter" (OSSL_DRBG_PARAM_RESEED_COUNTER) <unsigned integer>

These parameters work as described in "PARAMETERS" in EVP_RAND(3), except that they can all be set as well as read.

"test_entropy" (OSSL_RAND_PARAM_TEST_ENTROPY) <octet string>

Sets the bytes returned when the test generator is sent an entropy request. The current position is remembered across generate calls. If there are insufficient data present to satisfy a call, an error is returned.

"test_nonce" (OSSL_RAND_PARAM_TEST_NONCE) <octet string>

Sets the bytes returned when the test generator is sent a nonce request. Each nonce request will return all of the bytes.

NOTES

A context for a test generator can be obtained by calling:

```
EVP_RAND *rand = EVP_RAND_fetch(NULL, "TEST-RAND", NULL);
EVP_RAND_CTX *rctx = EVP_RAND_CTX_new(rand);
```

EXAMPLES

```
EVP_RAND *rand;
EVP_RAND_CTX *rctx;
unsigned char bytes[100];
OSSL_PARAM params[4], *p = params;
unsigned char entropy[1000] = { ... };
unsigned char nonce[20] = { ... };
```

```
unsigned int strength = 48;

rand = EVP_RAND_fetch(NULL, "TEST-RAND", NULL);
rctx = EVP_RAND_CTX_new(rand, NULL);
EVP_RAND_free(rand);

*p++ = OSSL_PARAM_construct_uint(OSSL_RAND_PARAM_STRENGTH, &strength);
*p++ = OSSL_PARAM_construct_octet_string(OSSL_RAND_PARAM_TEST_ENTROPY,
                                         entropy, sizeof(entropy));
*p++ = OSSL_PARAM_construct_octet_string(OSSL_RAND_PARAM_TEST_NONCE,
                                         nonce, sizeof(nonce));
*p = OSSL_PARAM_construct_end();
EVP_RAND_instantiate(rctx, strength, 0, NULL, 0, params);

EVP_RAND_generate(rctx, bytes, sizeof(bytes), strength, 0, NULL, 0);

EVP_RAND_CTX_free(rctx);
```

SEE ALSO

EVP_RAND(3), "PARAMETERS" in EVP_RAND(3)

COPYRIGHT