NAME

SRP_create_verifier_ex, SRP_create_verifier, SRP_create_verifier_BN_ex, SRP_create_verifier_BN, SRP_check_known_gN_param, SRP_get_default_gN - SRP authentication primitives

SYNOPSIS

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

The following functions have been deprecated since OpenSSL 3.0, and can be hidden entirely by defining **OPENSSL_API_COMPAT** with a suitable version value, see **openssl_user_macros**(7):

```
int SRP_create_verifier_BN_ex(const char *user, const char *pass, BIGNUM **salt,
                              BIGNUM **verifier, const BIGNUM *N,
                              const BIGNUM *g, OSSL_LIB_CTX *libctx,
                              const char *propq);
char *SRP_create_verifier_BN(const char *user, const char *pass, BIGNUM **salt,
                             BIGNUM **verifier, const BIGNUM *N, const BIGNUM *g);
char *SRP_create_verifier_ex(const char *user, const char *pass, char **salt,
                             char **verifier, const char *N, const char *g,
                             OSSL_LIB_CTX *libctx, const char *propq);
char *SRP_create_verifier(const char *user, const char *pass, char **salt,
                          char **verifier, const char *N, const char *g);

char *SRP_check_known_gN_param(const BIGNUM *g, const BIGNUM *N);
SRP_gN *SRP_get_default_gN(const char *id);
```

DESCRIPTION

All of the functions described on this page are deprecated. There are no available replacement functions at this time.

The **SRP_create_verifier_BN_ex()** function creates an SRP password verifier from the supplied parameters as defined in section 2.4 of RFC 5054 using the library context *libctx* and property query string *propq*. Any cryptographic algorithms that need to be fetched will use the *libctx* and *propq*. See "ALGORITHM FETCHING" in **crypto**(7).

SRP_create_verifier_BN() is the same as **SRP_create_verifier_BN_ex()** except the default library context and property query string is used.

On successful exit **verifier* will point to a newly allocated BIGNUM containing the verifier and (if a salt was not provided) **salt* will be populated with a newly allocated BIGNUM containing a random salt. If **salt* is not NULL then the provided salt is used instead. The caller is responsible for freeing the

allocated **salt* and **verifier* BIGNUMS (use **BN_free**(3)).

The **SRP_create_verifier()** function is similar to **SRP_create_verifier_BN()** but all numeric parameters are in a non-standard base64 encoding originally designed for compatibility with libsrp. This is mainly present for historical compatibility and its use is discouraged. It is possible to pass NULL as *N* and an SRP group id as *g* instead to load the appropriate gN values (see **SRP_get_default_gN()**). If both *N* and *g* are NULL the 8192-bit SRP group parameters are used. The caller is responsible for freeing the allocated **salt* and **verifier* (use **OPENSSL_free**(3)).

The **SRP_check_known_gN_param()** function checks that *g* and *N* are valid SRP group parameters from RFC 5054 appendix A.

The **SRP_get_default_gN()** function returns the gN parameters for the RFC 5054 *id* SRP group size. The known ids are "1024", "1536", "2048", "3072", "4096", "6144" and "8192".

RETURN VALUES

SRP_create_verifier_BN_ex() and **SRP_create_verifier_BN()** return 1 on success and 0 on failure.

SRP_create_verifier_ex() and **SRP_create_verifier()** return NULL on failure and a non-NULL value on success: "*" if *N* is not NULL, the selected group id otherwise. This value should not be freed.

SRP_check_known_gN_param() returns the text representation of the group id (i.e. the prime bit size) or NULL if the arguments are not valid SRP group parameters. This value should not be freed.

SRP_get_default_gN() returns NULL if *id* is not a valid group size, or the 8192-bit group parameters if *id* is NULL.

EXAMPLES

Generate and store a 8192 bit password verifier (error handling omitted for clarity):

```
#include <openssl/bn.h>
#include <openssl/srp.h>

const char *username = "username";
const char *password = "password";

SRP_VBASE *srpData = SRP_VBASE_new(NULL);

SRP_gN *gN = SRP_get_default_gN("8192");
```

```
BIGNUM *salt = NULL, *verifier = NULL;
SRP_create_verifier_BN_ex(username, password, &salt, &verifier, gN->N, gN->g,
                          NULL, NULL);

SRP_user_pwd *pwd = SRP_user_pwd_new();
SRP_user_pwd_set1_ids(pwd, username, NULL);
SRP_user_pwd_set0_sv(pwd, salt, verifier);
SRP_user_pwd_set_gN(pwd, gN->g, gN->N);

SRP_VBASE_add0_user(srpData, pwd);
```

SEE ALSO

openssl-srp(1), **SRP_VBASE_new**(3), **SRP_user_pwd_new**(3)

HISTORY

SRP_create_verifier_BN_ex() and **SRP_create_verifier_ex()** were introduced in OpenSSL 3.0. All other functions were added in OpenSSL 1.0.1.

All of these functions were deprecated in OpenSSL 3.0.

COPYRIGHT