NAME

EVP_PKEY−DH, EVP_PKEY−DHX, EVP_KEYMGMT−DH − EVP_PKEY DH and DHX keytype and algorithm support

DESCRIPTION

For **DH** FFC key agreement, two classes of domain parameters can be used: ‘‘safe’’ domain parameters that are associated with approved named safe-prime groups, and a class of ‘‘FIPS 186−type’’ domain parameters. FIPS 186−type domain parameters should only be used for backward compatibility with existing applications that cannot be upgraded to use the approved safe-prime groups.

See **EVP_PKEY−FFC** (7) for more information about FFC keys.

The **DH** key type uses PKCS#3 format which saves p and g, but not the ’q’ value. The **DHX** key type uses X9.42 format which saves the value of ’q’ and this must be used for FIPS186−4.

For **DHX** that is not a named group the FIPS186−4 standard specifies that the values used for FFC parameter generation are also required for parameter validation. This means that optional FFC domain parameter values for *seed*, *pcounter* and *gindex* or *hindex* may need to be stored for validation purposes. For **DHX** the *seed* and *pcounter* can be stored in ASN1 data (but the *gindex* or *hindex* can not be stored).

DH and DHX domain parameters

In addition to the common FCC parameters that all FFC keytypes should support (see ‘‘FFC parameters’’ in **EVP_PKEY−FFC** (7)) the **DHX** and **DH** keytype implementations support the following:

‘‘group’’ (**OSSL_PKEY_PARAM_GROUP_NAME**) <UTF8 string>

Sets or gets a string that associates a **DH** or **DHX** named safe prime group with known values for *p*, *q* and *g*.

The following values can be used by the OpenSSL’s default and FIPS providers: ‘‘ffdhe2048’’, ‘‘ffdhe3072’’, ‘‘ffdhe4096’’, ‘‘ffdhe6144’’, ‘‘ffdhe8192’’, ‘‘modp_2048’’, ‘‘modp_3072’’, ‘‘modp_4096’’, ‘‘modp_6144’’, ‘‘modp_8192’’.

The following additional values can also be used by OpenSSL’s default provider: ‘‘modp_1536’’, ‘‘dh_1024_160’’, ‘‘dh_2048_224’’, ‘‘dh_2048_256’’.

DH/DHX named groups can be easily validated since the parameters are well known. For protocols that only transfer *p* and *g* the value of *q* can also be retrieved.

DH and DHX additional parameters

‘‘encoded-pub-key’’ (**OSSL_PKEY_PARAM_ENCODED_PUBLIC_KEY**) <octet string>

Used for getting and setting the encoding of the DH public key used in a key exchange message for the TLS protocol. See **EVP_PKEY_set1_encoded_public_key()** and **EVP_PKEY_get1_encoded_public_key()**.

DH additional domain parameters

‘‘safeprime-generator’’ (**OSSL_PKEY_PARAM_DH_GENERATOR**) <integer>

Used for DH generation of safe primes using the old safe prime generator code. The default value is 2. It is recommended to use a named safe prime group instead, if domain parameter validation is required.

Randomly generated safe primes are not allowed by FIPS, so setting this value for the OpenSSL FIPS provider will instead choose a named safe prime group based on the size of *p*.

DH and DHX domain parameter / key generation parameters

In addition to the common FFC key generation parameters that all FFC key types should support (see ‘‘FFC key generation parameters’’ in **EVP_PKEY−FFC** (7)) the **DH** and **DHX** keytype implementation supports the following:

‘‘type’’ (**OSSL_PKEY_PARAM_FFC_TYPE**) <UTF8 string>

Sets the type of parameter generation. For **DH** valid values are:

“fips186_4”
“default”
“fips186_2”
These are described in “FFC key generation parameters” in **EVP_PKEY−FFC** (7)

“group”
This specifies that a named safe prime name will be chosen using the “pbits” type.

“generator”
A safe prime generator. See the “safeprime-generator” type above. This is only valid for **DH** keys.

“pbits” (**OSSL_PKEY_PARAM_FFC_PBITS**) <unsigned integer>
Sets the size (in bits) of the prime ’p’.

For “fips186_4” this must be 2048. For “fips186_2” this must be 1024. For “group” this can be any one of 2048, 3072, 4096, 6144 or 8192.

“priv_len” (**OSSL_PKEY_PARAM_DH_PRIV_LEN**) <integer>
An optional value to set the maximum length of the generated private key. The default value used if this is not set is the maximum value of BN_num_bits(*q*)). The minimum value that this can be set to is 2 * s. Where s is the security strength of the key which has values of 112, 128, 152, 176 and 200 for key sizes of 2048, 3072, 4096, 6144 and 8192.

EXAMPLES

An **EVP_PKEY** context can be obtained by calling:

```
EVP_PKEY_CTX *pctx = EVP_PKEY_CTX_new_from_name(NULL, "DH", NULL);
```

A **DH** key can be generated with a named safe prime group by calling:

```
int priv_len = 2 * 112;
OSSL_PARAM params[3];
EVP_PKEY *pkey = NULL;
EVP_PKEY_CTX *pctx = EVP_PKEY_CTX_new_from_name(NULL, "DH", NULL);

params[0] = OSSL_PARAM_construct_utf8_string("group", "ffdhe2048", 0);
/* "priv_len" is optional */
params[1] = OSSL_PARAM_construct_int("priv_len", &priv_len);
params[2] = OSSL_PARAM_construct_end();

EVP_PKEY_keygen_init(pctx);
EVP_PKEY_CTX_set_params(pctx, params);
EVP_PKEY_generate(pctx, &pkey);
...
EVP_PKEY_free(pkey);
EVP_PKEY_CTX_free(pctx);
```

DHX domain parameters can be generated according to **FIPS 186−4** by calling:

```
int gindex = 2;
unsigned int pbits = 2048;
unsigned int qbits = 256;
OSSL_PARAM params[6];
EVP_PKEY *param_key = NULL;
EVP_PKEY_CTX *pctx = NULL;

pctx = EVP_PKEY_CTX_new_from_name(NULL, "DHX", NULL);
EVP_PKEY_paramgen_init(pctx);

params[0] = OSSL_PARAM_construct_uint("pbits", &pbits);
```

```
params[1] = OSSL_PARAM_construct_uint("qbits", &qbits);
params[2] = OSSL_PARAM_construct_int("gindex", &gindex);
params[3] = OSSL_PARAM_construct_utf8_string("type", "fips186_4", 0);
params[4] = OSSL_PARAM_construct_utf8_string("digest", "SHA256", 0);
params[5] = OSSL_PARAM_construct_end();
EVP_PKEY_CTX_set_params(pctx, params);

EVP_PKEY_generate(pctx, &param_key);

EVP_PKEY_print_params(bio_out, param_key, 0, NULL);
...
EVP_PKEY_free(param_key);
EVP_PKEY_CTX_free(pctx);
```

A **DH** key can be generated using domain parameters by calling:

```
EVP_PKEY *key = NULL;
EVP_PKEY_CTX *gctx = EVP_PKEY_CTX_new_from_pkey(NULL, param_key, NULL);

EVP_PKEY_keygen_init(gctx);
EVP_PKEY_generate(gctx, &key);
EVP_PKEY_print_private(bio_out, key, 0, NULL);
...
EVP_PKEY_free(key);
EVP_PKEY_CTX_free(gctx);
```

To validate **FIPS 186−4 DHX** domain parameters decoded from **PEM** or **DER** data, additional values used during generation may be required to be set into the key.

EVP_PKEY_todata(), **OSSL_PARAM_merge()**, and **EVP_PKEY_fromdata()** are useful to add these parameters to the original key or domain parameters before the actual validation. In production code the return values should be checked.

```
EVP_PKEY *received_domp = ...; /* parameters received and decoded */
unsigned char *seed = ...;     /* and additional parameters received */
size_t seedlen = ...;          /* by other means, required */
int gindex = ...;              /* for the validation */
int pcounter = ...;
int hindex = ...;
OSSL_PARAM extra_params[4];
OSSL_PARAM *domain_params = NULL;
OSSL_PARAM *merged_params = NULL;
EVP_PKEY_CTX *ctx = NULL, *validate_ctx = NULL;
EVP_PKEY *complete_domp = NULL;

EVP_PKEY_todata(received_domp, OSSL_KEYMGMT_SELECT_DOMAIN_PARAMETERS,
                &domain_params);
extra_params[0] = OSSL_PARAM_construct_octet_string("seed", seed, seedlen);
/*
 * NOTE: For unverifiable g use "hindex" instead of "gindex"
 * extra_params[1] = OSSL_PARAM_construct_int("hindex", &hindex);
 */
extra_params[1] = OSSL_PARAM_construct_int("gindex", &gindex);
extra_params[2] = OSSL_PARAM_construct_int("pcounter", &pcounter);
extra_params[3] = OSSL_PARAM_construct_end();
merged_params = OSSL_PARAM_merge(domain_params, extra_params);
```

```
ctx = EVP_PKEY_CTX_new_from_name(NULL, "DHX", NULL);
EVP_PKEY_fromdata_init(ctx);
EVP_PKEY_fromdata(ctx, &complete_domp, OSSL_KEYMGMT_SELECT_ALL,
                  merged_params);

validate_ctx = EVP_PKEY_CTX_new_from_pkey(NULL, complete_domp, NULL);
if (EVP_PKEY_param_check(validate_ctx) > 0)
    /* validation_passed(); */
else
    /* validation_failed(); */

OSSL_PARAM_free(domain_params);
OSSL_PARAM_free(merged_params);
EVP_PKEY_CTX_free(ctx);
EVP_PKEY_CTX_free(validate_ctx);
EVP_PKEY_free(complete_domp);
```

CONFORMING TO

RFC 7919 (TLS ffdhe named safe prime groups)
RFC 3526 (IKE modp named safe prime groups)
RFC 5114 (Additional DH named groups for dh_1024_160‘‘, ’’dh_2048_224‘‘ and ’’dh_2048_256").

The following sections of SP800−56Ar3:

5.5.1.1 FFC Domain Parameter Selection/Generation
Appendix D: FFC Safe-prime Groups

The following sections of FIPS 186−4:

A.1.1.2 Generation of Probable Primes p and q Using an Approved Hash Function.
A.2.3 Generation of canonical generator g.
A.2.1 Unverifiable Generation of the Generator g.

SEE ALSO

EVP_PKEY−FFC (7), **EVP_KEYEXCH−DH** (7) **EVP_PKEY** (3), **provider−keymgmt** (7), **EVP_KEYMGMT** (3), **OSSL_PROVIDER−default** (7), **OSSL_PROVIDER−FIPS** (7)

COPYRIGHT