NAME

EVP_KEYEXCH−ECDH − ECDH Key Exchange algorithm support

DESCRIPTION

Key exchange support for the **ECDH** key type.

ECDH Key Exchange parameters

‘‘ecdh-cofactor-mode’’ (**OSSL_EXCHANGE_PARAM_EC_ECDH_COFACTOR_MODE**) <integer>

Sets or gets the ECDH mode of operation for the associated key exchange ctx.

In the context of an Elliptic Curve Diffie-Hellman key exchange, this parameter can be used to select between the plain Diffie-Hellman (DH) or Cofactor Diffie-Hellman (CDH) variants of the key exchange algorithm.

When setting, the value should be 1, 0 or −1, respectively forcing cofactor mode on, off, or resetting it to the default for the private key associated with the given key exchange ctx.

When getting, the value should be either 1 or 0, respectively signaling if the cofactor mode is on or off.

See also **provider−keymgmt** (7) for the related **OSSL_PKEY_PARAM_USE_COFACTOR_ECDH** parameter that can be set on a per-key basis.

‘‘kdf-type’’ (**OSSL_EXCHANGE_PARAM_KDF_TYPE**) <UTF8 string>

See ‘‘Common Key Exchange parameters’’ in **provider−keyexch** (7).

‘‘kdf-digest’’ (**OSSL_EXCHANGE_PARAM_KDF_DIGEST**) <UTF8 string>

See ‘‘Common Key Exchange parameters’’ in **provider−keyexch** (7).

‘‘kdf-digest-props’’ (**OSSL_EXCHANGE_PARAM_KDF_DIGEST_PROPS**) <UTF8 string>

See ‘‘Common Key Exchange parameters’’ in **provider−keyexch** (7).

‘‘kdf-outlen’’ (**OSSL_EXCHANGE_PARAM_KDF_OUTLEN**) <unsigned integer>

See ‘‘Common Key Exchange parameters’’ in **provider−keyexch** (7).

‘‘kdf-ukm’’ (**OSSL_EXCHANGE_PARAM_KDF_UKM**) <octet string>

See ‘‘Common Key Exchange parameters’’ in **provider−keyexch** (7).

EXAMPLES

Keys for the host and peer must be generated as shown in ‘‘Examples’’ in **EVP_PKEY−EC** (7) using the same curve name.

The code to generate a shared secret for the normal case is identical to ‘‘Examples’’ in **EVP_KEYEXCH−DH** (7).

To derive a shared secret on the host using the host’s key and the peer’s public key but also using X963KDF with a user key material:

```
/* It is assumed that the host_key, peer_pub_key and ukm are set up */
void derive_secret(EVP_PKEY *host_key, EVP_PKEY *peer_key,
                   unsigned char *ukm, size_t ukm_len)
{
    unsigned char secret[64];
    size_t out_len = sizeof(secret);
    size_t secret_len = out_len;
    unsigned int pad = 1;
    OSSL_PARAM params[6];
    EVP_PKEY_CTX *dctx = EVP_PKEY_CTX_new_from_pkey(NULL, host_key, NULL);

    EVP_PKEY_derive_init(dctx);

    params[0] = OSSL_PARAM_construct_uint(OSSL_EXCHANGE_PARAM_PAD, &pad);
    params[1] = OSSL_PARAM_construct_utf8_string(OSSL_EXCHANGE_PARAM_KDF_TYPE
                                                 "X963KDF", 0);
```

```
            params[2] = OSSL_PARAM_construct_utf8_string(OSSL_EXCHANGE_PARAM_KDF_DIGES
                                                         "SHA1", 0);
            params[3] = OSSL_PARAM_construct_size_t(OSSL_EXCHANGE_PARAM_KDF_OUTLEN,
                                                    &out_len);
            params[4] = OSSL_PARAM_construct_octet_string(OSSL_EXCHANGE_PARAM_KDF_UKM
                                                          ukm, ukm_len);
            params[5] = OSSL_PARAM_construct_end();
            EVP_PKEY_CTX_set_params(dctx, params);

            EVP_PKEY_derive_set_peer(dctx, peer_pub_key);
            EVP_PKEY_derive(dctx, secret, &secret_len);
            ...
            OPENSSL_clear_free(secret, secret_len);
            EVP_PKEY_CTX_free(dctx);
        }
```

SEE ALSO

EVP_PKEY−EC (7) **EVP_PKEY** (3), **provider−keyexch** (7), **provider−keymgmt** (7), **OSSL_PROVIDER−default** (7), **OSSL_PROVIDER−FIPS** (7),

COPYRIGHT