NAME

EVP_SIGNATURE−ED25519, EVP_SIGNATURE−ED448, Ed25519, Ed448 − The EVP_PKEY Ed25519 and Ed448 signature implementations

DESCRIPTION

The **Ed25519** and **Ed448** EVP_PKEY implementation supports key generation, one−shot digest−sign and digest−verify using the EdDSA signature schemes described in RFC 8032. It has associated private and public key formats compatible with RFC 8410.

EdDSA Instances

RFC 8032 describes five EdDSA instances: Ed25519, Ed25519ctx, Ed25519ph, Ed448, Ed448ph.

The instances Ed25519, Ed25519ctx, Ed448 are referred to as **PureEdDSA** schemes. For these three instances, the sign and verify procedures require access to the complete message (not a digest of the message).

The instances Ed25519ph, Ed448ph are referred to as **HashEdDSA** schemes. For these two instances, the sign and verify procedures do not require access to the complete message; they operate on a hash of the message. For Ed25519ph, the hash function is SHA512. For Ed448ph, the hash function is SHAKE256 with an output length of 512 bits.

The instances Ed25519ctx, Ed25519ph, Ed448, Ed448ph accept an optional **context−string** as input to sign and verify operations (and for Ed25519ctx, the context−string must be nonempty). For the Ed25519 instance, a nonempty context−string is not permitted.

These instances can be specified as signature parameters when using **EVP_DigestSignInit**(3) and **EVP_DigestVerifyInit**(3), see "ED25519 and ED448 Signature Parameters" below.

These instances are also explicitly fetchable as algorithms using **EVP_SIGNATURE_fetch**(3), which can be used with **EVP_PKEY_sign_init_ex2**(3), **EVP_PKEY_verify_init_ex2**(3), **EVP_PKEY_sign_message_init**(3) and **EVP_PKEY_verify_message_init**(3).

ED25519 and ED448 Signature Parameters

Two parameters can be set during signing or verification: the EdDSA **instance name** and the **context−string value**. They can be set by passing an OSSL_PARAM array to **EVP_DigestSignInit_ex()**.

- "instance" (**OSSL_SIGNATURE_PARAM_INSTANCE**) <utf8 string>

  One of the five strings "Ed25519", "Ed25519ctx", "Ed25519ph", "Ed448", "Ed448ph".

  "Ed25519", "Ed25519ctx", "Ed25519ph" are valid only for an Ed25519 EVP_PKEY.

  "Ed448", "Ed448ph" are valid only for an Ed448 EVP_PKEY.

- "context−string" (**OSSL_SIGNATURE_PARAM_CONTEXT_STRING**) <octet string>

  A string of octets with length at most 255.

Both of these parameters are optional.

When using **EVP_DigestSignInit**(3) or **EVP_DigestVerifyInit**(3), the signature algorithm is derived from the key type name. The key type name ("Ed25519" or "Ed448") is also the default for the instance, but this can be changed with the "instance" parameter.

Note that a message digest name must **NOT** be specified when signing or verifying.

When using **EVP_PKEY_sign_init_ex2**(3), **EVP_PKEY_verify_init_ex2**(3), **EVP_PKEY_sign_message_init**(3) or **EVP_PKEY_verify_message_init**(3), the instance is the explicit signature algorithm name, and may not be changed (trying to give one with the "instance" parameter is therefore an error).

If a context−string is not specified, then an empty context−string is used.

See **EVP_PKEY−X25519**(7) for information related to **X25519** and **X448** keys.

The following signature parameters can be retrieved using **EVP_PKEY_CTX_get_params()**.

- "algorithm−id" (**OSSL_SIGNATURE_PARAM_ALGORITHM_ID**) <octet string>
- "instance" (**OSSL_SIGNATURE_PARAM_INSTANCE**) <utf8 string>
- "context−string" (**OSSL_SIGNATURE_PARAM_CONTEXT_STRING**) <octet string>

The parameters are described in **provider−signature**(7).

NOTES

The PureEdDSA instances do not support the streaming mechanism of other signature algorithms using, for example, **EVP_DigestUpdate()**. The message to sign or verify must be passed using the one−shot **EVP_DigestSign()** and **EVP_DigestVerify()** functions.

The HashEdDSA instances do not yet support the streaming mechanisms (so the one−shot functions must be used with HashEdDSA as well).

When calling **EVP_DigestSignInit()** or **EVP_DigestVerifyInit()**, the digest *type* parameter **MUST** be set to NULL.

Applications wishing to sign certificates (or other structures such as CRLs or certificate requests) using Ed25519 or Ed448 can either use **X509_sign()** or **X509_sign_ctx()** in the usual way.

Ed25519 or Ed448 private keys can be set directly using **EVP_PKEY_new_raw_private_key**(3) or loaded from a PKCS#8 private key file using **PEM_read_bio_PrivateKey**(3) (or similar function). Completely new keys can also be generated (see the example below). Setting a private key also sets the associated public key.

Ed25519 or Ed448 public keys can be set directly using **EVP_PKEY_new_raw_public_key**(3) or loaded from a SubjectPublicKeyInfo structure in a PEM file using **PEM_read_bio_PUBKEY**(3) (or similar function).

Ed25519 and Ed448 can be tested with the **openssl−speed**(1) application since version 1.1.1. Valid algorithm names are **ed25519**, **ed448** and **eddsa**. If **eddsa** is specified, then both Ed25519 and Ed448 are benchmarked.

Since Ed25519ctx is not included in FIPS 186−5, it is not present in the FIPS provider.

EXAMPLES

To sign a message using an ED25519 EVP_PKEY structure:

```
void do_sign(EVP_PKEY *ed_key, unsigned char *msg, size_t msg_len)
{
    size_t sig_len;
    unsigned char *sig = NULL;
    EVP_MD_CTX *md_ctx = EVP_MD_CTX_new();

    const OSSL_PARAM params[] = {
        OSSL_PARAM_utf8_string ("instance", "Ed25519ctx", 10),
        OSSL_PARAM_octet_string("context−string", (unsigned char *)"A protocol d
        OSSL_PARAM_END
    };

    /* The input "params" is not needed if default options are acceptable.
       Use NULL in place of "params" in that case. */
    EVP_DigestSignInit_ex(md_ctx, NULL, NULL, NULL, NULL, ed_key, params);
    /* Calculate the required size for the signature by passing a NULL buffer. *
    EVP_DigestSign(md_ctx, NULL, &sig_len, msg, msg_len);
    sig = OPENSSL_zalloc(sig_len);

    EVP_DigestSign(md_ctx, sig, &sig_len, msg, msg_len);
    ...
    OPENSSL_free(sig);
```

```
        EVP_MD_CTX_free(md_ctx);
    }
```

SEE ALSO

EVP_PKEY−X25519 (7) **provider−signature** (7), **EVP_DigestSignInit** (3), **EVP_DigestVerifyInit** (3),

COPYRIGHT