

NAME

Ed25519, Ed448 – EVP_PKEY Ed25519 and Ed448 support

DESCRIPTION

The **Ed25519** and **Ed448** EVP_PKEY implementation supports key generation, one-shot digest sign and digest verify using PureEdDSA and **Ed25519** or **Ed448** (see RFC8032). It has associated private and public key formats compatible with draft-ietf-curdle-pkix-04.

No additional parameters can be set during key generation, one-shot signing or verification. In particular, because PureEdDSA is used, a digest must **NOT** be specified when signing or verifying.

NOTES

The PureEdDSA algorithm does 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.

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.

A context for the **Ed25519** algorithm can be obtained by calling:

```
EVP_PKEY_CTX *pctx = EVP_PKEY_CTX_new_id(EVP_PKEY_ED25519, NULL);
```

For the **Ed448** algorithm a context can be obtained by calling:

```
EVP_PKEY_CTX *pctx = EVP_PKEY_CTX_new_id(EVP_PKEY_ED448, NULL);
```

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 within **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.

EXAMPLE

This example generates an **Ed25519** private key and writes it to standard output in PEM format:

```
#include <openssl/evp.h>
#include <openssl/pem.h>
...
EVP_PKEY *pkey = NULL;
EVP_PKEY_CTX *pctx = EVP_PKEY_CTX_new_id(EVP_PKEY_ED25519, NULL);
EVP_PKEY_keygen_init(pctx);
EVP_PKEY_keygen(pctx, &pkey);
EVP_PKEY_CTX_free(pctx);
PEM_write_PrivateKey(stdout, pkey, NULL, NULL, 0, NULL, NULL);
```

SEE ALSO

EVP_PKEY_CTX_new(3), **EVP_PKEY_keygen(3)**, **EVP_DigestSignInit(3)**,
EVP_DigestVerifyInit(3),

COPYRIGHT

Copyright 2017–2018 The OpenSSL Project Authors. All Rights Reserved.

Licensed under the OpenSSL license (the “License”). You may not use this file except in compliance with the License. You can obtain a copy in the file **LICENSE** in the source distribution or at <https://www.openssl.org/source/license.html>.