NAME

X25519, X448 − EVP_PKEY X25519 and X448 support

DESCRIPTION

The **X25519** and **X448** EVP_PKEY implementation supports key generation and key derivation using **X25519** and **X448**. It has associated private and public key formats compatible with RFC 8410.

No additional parameters can be set during key generation.

The peer public key must be set using **EVP_PKEY_derive_set_peer()** when performing key derivation.

NOTES

A context for the **X25519** algorithm can be obtained by calling:

```
 EVP_PKEY_CTX *pctx = EVP_PKEY_CTX_new_id(EVP_PKEY_X25519, NULL);
```

For the **X448** algorithm a context can be obtained by calling:

```
 EVP_PKEY_CTX *pctx = EVP_PKEY_CTX_new_id(EVP_PKEY_X448, NULL);
```

X25519 or X448 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.

X25519 or X448 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).

EXAMPLES

This example generates an **X25519** 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_X25519, 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);
```

The key derivation example in **EVP_PKEY_derive**(3) can be used with **X25519** and **X448**.

SEE ALSO

EVP_PKEY_CTX_new(3), **EVP_PKEY_keygen**(3), **EVP_PKEY_derive**(3), **EVP_PKEY_derive_set_peer**(3)

COPYRIGHT