NAME

ossl_store − Store retrieval functions

SYNOPSIS

#include <openssl/store.h>

DESCRIPTION

General

A STORE is a layer of functionality to retrieve a number of supported objects from a repository of any kind, addressable as a filename or as a URI.

The functionality supports the pattern ‘‘open a channel to the repository’’, ‘‘loop and retrieve one object at a time’’, and ‘‘finish up by closing the channel’’.

The retrieved objects are returned as a wrapper type **OSSL_STORE_INFO**, from which an OpenSSL type can be retrieved.

URI schemes and loaders

Support for a URI scheme is called a STORE ‘‘loader’’, and can be added dynamically from the calling application or from a loadable engine.

Support for the ’file’ scheme is built into `libcrypto`. See**ossl_stor e−file** (7) for more information.

UI_METHOD and pass phrases

The **OSS_STORE** API does nothing to enforce any specific format or encoding on the pass phrase that the **UI_METHOD** provides. However, the pass phrase is expected to be UTF−8 encoded. The result of any other encoding is undefined.

EXAMPLES

A generic call

```
OSSL_STORE_CTX *ctx = OSSL_STORE_open("file:/foo/bar/data.pem");

/*
 * OSSL_STORE_eof() simulates file semantics for any repository to signal
 * that no more data can be expected
 */
while (!OSSL_STORE_eof(ctx)) {
    OSSL_STORE_INFO *info = OSSL_STORE_load(ctx);

    /*
     * Do whatever is necessary with the OSSL_STORE_INFO,
     * here just one example
     */
    switch (OSSL_STORE_INFO_get_type(info)) {
    case OSSL_STORE_INFO_CERT:
        /* Print the X.509 certificate text */
        X509_print_fp(stdout, OSSL_STORE_INFO_get0_CERT(info));
        /* Print the X.509 certificate PEM output */
        PEM_write_X509(stdout, OSSL_STORE_INFO_get0_CERT(info));
        break;
    }
}

OSSL_STORE_close(ctx);
```

SEE ALSO

OSSL_STORE_INFO (3), **OSSL_STORE_LOADER** (3), **OSSL_STORE_open** (3), **OSSL_STORE_expect** (3), **OSSL_STORE_SEARCH** (3)

COPYRIGHT