NAME

OSSL_STORE_INFO, OSSL_STORE_INFO_get_type, OSSL_STORE_INFO_get0_NAME, OSSL_STORE_INFO_get0_NAME_description, OSSL_STORE_INFO_get0_PARAMS, OSSL_STORE_INFO_get0_PUBKEY, OSSL_STORE_INFO_get0_PKEY, OSSL_STORE_INFO_get0_CERT, OSSL_STORE_INFO_get0_CRL, OSSL_STORE_INFO_get1_NAME, OSSL_STORE_INFO_get1_NAME_description, OSSL_STORE_INFO_get1_PARAMS, OSSL_STORE_INFO_get1_PUBKEY, OSSL_STORE_INFO_get1_PKEY, OSSL_STORE_INFO_get1_CERT, OSSL_STORE_INFO_get1_CRL, OSSL_STORE_INFO_type_string, OSSL_STORE_INFO_free, OSSL_STORE_INFO_new_NAME, OSSL_STORE_INFO_set0_NAME_description, OSSL_STORE_INFO_new_PARAMS, OSSL_STORE_INFO_new_PUBKEY, OSSL_STORE_INFO_new_PKEY, OSSL_STORE_INFO_new_CERT, OSSL_STORE_INFO_new_CRL, OSSL_STORE_INFO_new, OSSL_STORE_INFO_get0_data - Functions to manipulate OSSL_STORE_INFO objects

SYNOPSIS

```
#include <openssl/store.h>

typedef struct ossl_store_info_st OSSL_STORE_INFO;

int OSSL_STORE_INFO_get_type(const OSSL_STORE_INFO *store_info);
const char *OSSL_STORE_INFO_get0_NAME(const OSSL_STORE_INFO *store_info);
char *OSSL_STORE_INFO_get1_NAME(const OSSL_STORE_INFO *store_info);
const char *OSSL_STORE_INFO_get0_NAME_description(const OSSL_STORE_INFO
                                                  *store_info);
char *OSSL_STORE_INFO_get1_NAME_description(const OSSL_STORE_INFO *store_info);
EVP_PKEY *OSSL_STORE_INFO_get0_PARAMS(const OSSL_STORE_INFO *store_info);
EVP_PKEY *OSSL_STORE_INFO_get1_PARAMS(const OSSL_STORE_INFO *store_info);
EVP_PKEY *OSSL_STORE_INFO_get0_PUBKEY(const OSSL_STORE_INFO *info);
EVP_PKEY *OSSL_STORE_INFO_get1_PUBKEY(const OSSL_STORE_INFO *info);
EVP_PKEY *OSSL_STORE_INFO_get0_PKEY(const OSSL_STORE_INFO *store_info);
EVP_PKEY *OSSL_STORE_INFO_get1_PKEY(const OSSL_STORE_INFO *store_info);
X509 *OSSL_STORE_INFO_get0_CERT(const OSSL_STORE_INFO *store_info);
X509 *OSSL_STORE_INFO_get1_CERT(const OSSL_STORE_INFO *store_info);
X509_CRL *OSSL_STORE_INFO_get0_CRL(const OSSL_STORE_INFO *store_info);
X509_CRL *OSSL_STORE_INFO_get1_CRL(const OSSL_STORE_INFO *store_info);

const char *OSSL_STORE_INFO_type_string(int type);

void OSSL_STORE_INFO_free(OSSL_STORE_INFO *store_info);
```

```
OSSL_STORE_INFO *OSSL_STORE_INFO_new_NAME(char *name);
int OSSL_STORE_INFO_set0_NAME_description(OSSL_STORE_INFO *info, char *desc);
OSSL_STORE_INFO *OSSL_STORE_INFO_new_PARAMS(DSA *dsa_params);
OSSL_STORE_INFO *OSSL_STORE_INFO_new_PUBKEY(EVP_PKEY *pubkey);
OSSL_STORE_INFO *OSSL_STORE_INFO_new_PKEY(EVP_PKEY *pkey);
OSSL_STORE_INFO *OSSL_STORE_INFO_new_CERT(X509 *x509);
OSSL_STORE_INFO *OSSL_STORE_INFO_new_CRL(X509_CRL *crl);

OSSL_STORE_INFO *OSSL_STORE_INFO_new(int type, void *data);
void *OSSL_STORE_INFO_get0_data(int type, const OSSL_STORE_INFO *info);
```

DESCRIPTION

These functions are primarily useful for applications to retrieve supported objects from **OSSL_STORE_INFO** objects and for scheme specific loaders to create **OSSL_STORE_INFO** holders.

Types

OSSL_STORE_INFO is an opaque type that's just an intermediary holder for the objects that have been retrieved by **OSSL_STORE_load()** and similar functions. Supported OpenSSL type object can be extracted using one of STORE_INFO_get0_<TYPE>() where <TYPE> can be NAME, PARAMS, PKEY, CERT, or CRL. The life time of this extracted object is as long as the life time of the **OSSL_STORE_INFO** it was extracted from, so care should be taken not to free the latter too early. As an alternative, STORE_INFO_get1_<TYPE>() extracts a duplicate (or the same object with its reference count increased), which can be used after the containing **OSSL_STORE_INFO** has been freed. The object returned by STORE_INFO_get1_<TYPE>() must be freed separately by the caller. See "SUPPORTED OBJECTS" for more information on the types that are supported.

Functions

OSSL_STORE_INFO_get_type() takes a **OSSL_STORE_INFO** and returns the STORE type number for the object inside.

STORE_INFO_get_type_string() takes a STORE type number and returns a short string describing it.

OSSL_STORE_INFO_get0_NAME(), **OSSL_STORE_INFO_get0_NAME_description()**, **OSSL_STORE_INFO_get0_PARAMS()**, **OSSL_STORE_INFO_get0_PUBKEY()**, **OSSL_STORE_INFO_get0_PKEY()**, **OSSL_STORE_INFO_get0_CERT()**, **OSSL_STORE_INFO_get0_CRL()** all take a **OSSL_STORE_INFO** and return the object it holds if the **OSSL_STORE_INFO** type (as returned by **OSSL_STORE_INFO_get_type()**) matches the function, otherwise NULL.

OSSL_STORE_INFO_get1_NAME(), **OSSL_STORE_INFO_get1_NAME_description()**,

OSSL_STORE_INFO_get1_PARAMS(), **OSSL_STORE_INFO_get1_PUBKEY()**, **OSSL_STORE_INFO_get1_PKEY()**, **OSSL_STORE_INFO_get1_CERT()** and **OSSL_STORE_INFO_get1_CRL()** all take a **OSSL_STORE_INFO** and return a duplicate the object it holds if the **OSSL_STORE_INFO** type (as returned by **OSSL_STORE_INFO_get_type()**) matches the function, otherwise NULL.

OSSL_STORE_INFO_free() frees a **OSSL_STORE_INFO** and its contained type.

OSSL_STORE_INFO_new_NAME() , **OSSL_STORE_INFO_new_PARAMS()**, , **OSSL_STORE_INFO_new_PUBKEY()**, **OSSL_STORE_INFO_new_PKEY()**, **OSSL_STORE_INFO_new_CERT()** and **OSSL_STORE_INFO_new_CRL()** create a **OSSL_STORE_INFO** object to hold the given input object. On success the input object is consumed.

Additionally, for **OSSL_STORE_INFO_NAME** objects, **OSSL_STORE_INFO_set0_NAME_description()** can be used to add an extra description. This description is meant to be human readable and should be used for information printout.

OSSL_STORE_INFO_new() creates a **OSSL_STORE_INFO** with an arbitrary *type* number and *data* structure. It's the responsibility of the caller to define type numbers other than the ones defined by *<openssl/store.h>*, and to handle freeing the associated data structure on their own. *Using type numbers that are defined by <openssl/store.h> may cause undefined behaviours, including crashes.*

OSSL_STORE_INFO_get0_data() returns the data pointer that was passed to **OSSL_STORE_INFO_new()** if *type* matches the type number in *info*.

OSSL_STORE_INFO_new() and **OSSL_STORE_INFO_get0_data()** may be useful for applications that define their own STORE data, but must be used with care.

SUPPORTED OBJECTS

Currently supported object types are:

OSSL_STORE_INFO_NAME

A name is exactly that, a name. It's like a name in a directory, but formatted as a complete URI. For example, the path in URI "file:/foo/bar/" could include a file named "cookie.pem", and in that case, the returned **OSSL_STORE_INFO_NAME** object would have the URI "file:/foo/bar/cookie.pem", which can be used by the application to get the objects in that file. This can be applied to all schemes that can somehow support a listing of object URIs.

For "file:" URIs that are used without the explicit scheme, the returned name will be the path of each object, so if "/foo/bar" was given and that path has the file "cookie.pem", the name

"/foo/bar/cookie.pem" will be returned.

The returned URI is considered canonical and must be unique and permanent for the storage where the object (or collection of objects) resides. Each loader is responsible for ensuring that it only returns canonical URIs. However, it's possible that certain schemes allow an object (or collection thereof) to be reached with alternative URIs; just because one URI is canonical doesn't mean that other variants can't be used.

At the discretion of the loader that was used to get these names, an extra description may be attached as well.

OSSL_STORE_INFO_PARAMS
Key parameters.

OSSL_STORE_INFO_PKEY
A keypair or just a private key (possibly with key parameters).

OSSL_STORE_INFO_PUBKEY
A public key (possibly with key parameters).

OSSL_STORE_INFO_CERT
An X.509 certificate.

OSSL_STORE_INFO_CRL
A X.509 certificate revocation list.

RETURN VALUES

OSSL_STORE_INFO_get_type() returns the STORE type number of the given **OSSL_STORE_INFO**. There is no error value.

OSSL_STORE_INFO_get0_NAME(), **OSSL_STORE_INFO_get0_NAME_description()**, **OSSL_STORE_INFO_get0_PARAMS()**, **OSSL_STORE_INFO_get0_PKEY()**, **OSSL_STORE_INFO_get0_CERT()** and **OSSL_STORE_INFO_get0_CRL()** all return a pointer to the OpenSSL object on success, NULL otherwise.

OSSL_STORE_INFO_get1_NAME(), **OSSL_STORE_INFO_get1_NAME_description()**, **OSSL_STORE_INFO_get1_PARAMS()**, **OSSL_STORE_INFO_get1_PKEY()**, **OSSL_STORE_INFO_get1_CERT()** and **OSSL_STORE_INFO_get1_CRL()** all return a pointer to a duplicate of the OpenSSL object on success, NULL otherwise.

OSSL_STORE_INFO_type_string() returns a string on success, or NULL on failure.

OSSL_STORE_INFO_new_NAME(), **OSSL_STORE_INFO_new_PARAMS()**, **OSSL_STORE_INFO_new_PKEY()**, **OSSL_STORE_INFO_new_CERT()** and **OSSL_STORE_INFO_new_CRL()** return a **OSSL_STORE_INFO** pointer on success, or NULL on failure.

OSSL_STORE_INFO_set0_NAME_description() returns 1 on success, or 0 on failure.

SEE ALSO

ossl_store(7), **OSSL_STORE_open**(3), **OSSL_STORE_register_loader**(3)

HISTORY

The OSSL_STORE API was added in OpenSSL 1.1.1.

The OSSL_STORE_INFO_PUBKEY object type was added in OpenSSL 3.0.

COPYRIGHT