NAME

OSSL_STORE_LOADER, OSSL_STORE_LOADER_fetch, OSSL_STORE_LOADER_up_ref, OSSL_STORE_LOADER_free, OSSL_STORE_LOADER_get0_provider, OSSL_STORE_LOADER_get0_properties, OSSL_STORE_LOADER_is_a, OSSL_STORE_LOADER_get0_description, OSSL_STORE_LOADER_do_all_provided, OSSL_STORE_LOADER_names_do_all, OSSL_STORE_LOADER_CTX, OSSL_STORE_LOADER_new, OSSL_STORE_LOADER_get0_engine, OSSL_STORE_LOADER_get0_scheme, OSSL_STORE_LOADER_set_open, OSSL_STORE_LOADER_set_open_ex, OSSL_STORE_LOADER_set_attach, OSSL_STORE_LOADER_set_ctrl, OSSL_STORE_LOADER_set_expect, OSSL_STORE_LOADER_set_find, OSSL_STORE_LOADER_set_load, OSSL_STORE_LOADER_set_eof, OSSL_STORE_LOADER_set_error, OSSL_STORE_LOADER_set_close, OSSL_STORE_register_loader, OSSL_STORE_unregister_loader, OSSL_STORE_open_fn, OSSL_STORE_open_ex_fn, OSSL_STORE_attach_fn, OSSL_STORE_ctrl_fn, OSSL_STORE_expect_fn, OSSL_STORE_find_fn, OSSL_STORE_load_fn, OSSL_STORE_eof_fn, OSSL_STORE_error_fn, OSSL_STORE_close_fn - Types and functions to manipulate, register and unregister STORE loaders for different URI schemes

SYNOPSIS

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

typedef struct ossl_store_loader_st OSSL_STORE_LOADER;

OSSL_STORE_LOADER *OSSL_STORE_LOADER_fetch(OSSL_LIB_CTX *libctx,
                                           const char *scheme,
                                           const char *properties);
int OSSL_STORE_LOADER_up_ref(OSSL_STORE_LOADER *loader);
void OSSL_STORE_LOADER_free(OSSL_STORE_LOADER *loader);
const OSSL_PROVIDER *OSSL_STORE_LOADER_get0_provider(const OSSL_STORE_LOADER *
                                                     loader);
const char *OSSL_STORE_LOADER_get0_properties(const OSSL_STORE_LOADER *loader);
const char *OSSL_STORE_LOADER_get0_description(const OSSL_STORE_LOADER *loader);
int OSSL_STORE_LOADER_is_a(const OSSL_STORE_LOADER *loader,
                           const char *scheme);
void OSSL_STORE_LOADER_do_all_provided(OSSL_LIB_CTX *libctx,
                                       void (*user_fn)(OSSL_STORE_LOADER *loader,
                                                       void *arg),
                                       void *user_arg);
int OSSL_STORE_LOADER_names_do_all(const OSSL_STORE_LOADER *loader,
                                   void (*fn)(const char *name, void *data),
```

```
                    void *data);
```

The following functions have been deprecated since OpenSSL 3.0, and can be hidden entirely by defining **OPENSSL_API_COMPAT** with a suitable version value, see **openssl_user_macros**(7):

```
OSSL_STORE_LOADER *OSSL_STORE_LOADER_new(ENGINE *e, const char *scheme);
const ENGINE *OSSL_STORE_LOADER_get0_engine(const OSSL_STORE_LOADER
                                            *store_loader);
const char *OSSL_STORE_LOADER_get0_scheme(const OSSL_STORE_LOADER
                                          *store_loader);

/* struct ossl_store_loader_ctx_st is defined differently by each loader */
typedef struct ossl_store_loader_ctx_st OSSL_STORE_LOADER_CTX;

typedef OSSL_STORE_LOADER_CTX *(*OSSL_STORE_open_fn)(
    const char *uri, const UI_METHOD *ui_method, void *ui_data);
int OSSL_STORE_LOADER_set_open(OSSL_STORE_LOADER *store_loader,
                               OSSL_STORE_open_fn store_open_function);
typedef OSSL_STORE_LOADER_CTX *(*OSSL_STORE_open_ex_fn)(
    const char *uri, const UI_METHOD *ui_method, void *ui_data);
int OSSL_STORE_LOADER_set_open_ex
    (OSSL_STORE_LOADER *store_loader,
     OSSL_STORE_open_ex_fn store_open_ex_function);
typedef OSSL_STORE_LOADER_CTX *(*OSSL_STORE_attach_fn)
    (const OSSL_STORE_LOADER *loader, BIO *bio,
     OSSL_LIB_CTX *libctx, const char *propq,
     const UI_METHOD *ui_method, void *ui_data);
int OSSL_STORE_LOADER_set_attach(OSSL_STORE_LOADER *loader,
                                 OSSL_STORE_attach_fn attach_function);
typedef int (*OSSL_STORE_ctrl_fn)(OSSL_STORE_LOADER_CTX *ctx, int cmd,
                                  va_list args);
int OSSL_STORE_LOADER_set_ctrl(OSSL_STORE_LOADER *store_loader,
                               OSSL_STORE_ctrl_fn store_ctrl_function);
typedef int (*OSSL_STORE_expect_fn)(OSSL_STORE_LOADER_CTX *ctx, int expected);
int OSSL_STORE_LOADER_set_expect(OSSL_STORE_LOADER *loader,
                                 OSSL_STORE_expect_fn expect_function);
typedef int (*OSSL_STORE_find_fn)(OSSL_STORE_LOADER_CTX *ctx,
                                  OSSL_STORE_SEARCH *criteria);
int OSSL_STORE_LOADER_set_find(OSSL_STORE_LOADER *loader,
                               OSSL_STORE_find_fn find_function);
```

```
typedef OSSL_STORE_INFO *(*OSSL_STORE_load_fn)(OSSL_STORE_LOADER_CTX *ctx,
                                               UI_METHOD *ui_method,
                                               void *ui_data);
int OSSL_STORE_LOADER_set_load(OSSL_STORE_LOADER *store_loader,
                               OSSL_STORE_load_fn store_load_function);
typedef int (*OSSL_STORE_eof_fn)(OSSL_STORE_LOADER_CTX *ctx);
int OSSL_STORE_LOADER_set_eof(OSSL_STORE_LOADER *store_loader,
                              OSSL_STORE_eof_fn store_eof_function);
typedef int (*OSSL_STORE_error_fn)(OSSL_STORE_LOADER_CTX *ctx);
int OSSL_STORE_LOADER_set_error(OSSL_STORE_LOADER *store_loader,
                                OSSL_STORE_error_fn store_error_function);
typedef int (*OSSL_STORE_close_fn)(OSSL_STORE_LOADER_CTX *ctx);
int OSSL_STORE_LOADER_set_close(OSSL_STORE_LOADER *store_loader,
                                OSSL_STORE_close_fn store_close_function);
void OSSL_STORE_LOADER_free(OSSL_STORE_LOADER *store_loader);

int OSSL_STORE_register_loader(OSSL_STORE_LOADER *loader);
OSSL_STORE_LOADER *OSSL_STORE_unregister_loader(const char *scheme);
```

DESCRIPTION

OSSL_STORE_LOADER is a method for OSSL_STORE loaders, which implement **OSSL_STORE_open()**, **OSSL_STORE_open_ex()**, **OSSL_STORE_load()**, **OSSL_STORE_eof()**, **OSSL_STORE_error()** and **OSSL_STORE_close()** for specific storage schemes.

OSSL_STORE_LOADER_fetch() looks for an implementation for a storage *scheme* within the providers that has been loaded into the **OSSL_LIB_CTX** given by *libctx*, and with the properties given by *properties*.

OSSL_STORE_LOADER_up_ref() increments the reference count for the given *loader*.

OSSL_STORE_LOADER_free() decrements the reference count for the given *loader*, and when the count reaches zero, frees it.

OSSL_STORE_LOADER_get0_provider() returns the provider of the given *loader*.

OSSL_STORE_LOADER_get0_properties() returns the property definition associated with the given *loader*.

OSSL_STORE_LOADER_is_a() checks if *loader* is an implementation of an algorithm that’s identifiable with *scheme*.

OSSL_STORE_LOADER_get0_description() returns a description of the *loader*, meant for display and human consumption. The description is at the discretion of the *loader* implementation.

OSSL_STORE_LOADER_do_all_provided() traverses all store implementations by all activated providers in the library context *libctx*, and for each of the implementations, calls *user_fn* with the implementation method and *user_arg* as arguments.

OSSL_STORE_LOADER_names_do_all() traverses all names for the given *loader*, and calls *fn* with each name and *data*.

Legacy Types and Functions (deprecated)

These functions help applications and engines to create loaders for schemes they support. These are all deprecated and discouraged in favour of provider implementations, see **provider-storemgmt**(7).

OSSL_STORE_LOADER_CTX is a type template, to be defined by each loader using "struct ossl_store_loader_ctx_st { ... }".

OSSL_STORE_open_fn, **OSSL_STORE_open_ex_fn**, **OSSL_STORE_ctrl_fn**, **OSSL_STORE_expect_fn**, **OSSL_STORE_find_fn**, **OSSL_STORE_load_fn**, **OSSL_STORE_eof_fn**, and **OSSL_STORE_close_fn** are the function pointer types used within a STORE loader. The functions pointed at define the functionality of the given loader.

OSSL_STORE_open_fn and **OSSL_STORE_open_ex_fn**

OSSL_STORE_open_ex_fn takes a URI and is expected to interpret it in the best manner possible according to the scheme the loader implements. It also takes a **UI_METHOD** and associated data, to be used any time something needs to be prompted for, as well as a library context *libctx* with an associated property query *propq*, to be used when fetching necessary algorithms to perform the loads. Furthermore, this function is expected to initialize what needs to be initialized, to create a private data store (**OSSL_STORE_LOADER_CTX**, see above), and to return it. If something goes wrong, this function is expected to return NULL.

OSSL_STORE_open_fn does the same thing as **OSSL_STORE_open_ex_fn** but uses NULL for the library context *libctx* and property query *propq*.

OSSL_STORE_attach_fn

This function takes a **BIO**, otherwise works like **OSSL_STORE_open_ex_fn**.

OSSL_STORE_ctrl_fn

This function takes a **OSSL_STORE_LOADER_CTX** pointer, a command number *cmd* and a **va_list** *args* and is used to manipulate loader specific parameters.

Loader specific command numbers must begin at **OSSL_STORE_C_CUSTOM_START**. Any number below that is reserved for future globally known command numbers.

This function is expected to return 1 on success, 0 on error.

OSSL_STORE_expect_fn

This function takes a **OSSL_STORE_LOADER_CTX** pointer and a **OSSL_STORE_INFO** identity *expected*, and is used to tell the loader what object type is expected. *expected* may be zero to signify that no specific object type is expected.

This function is expected to return 1 on success, 0 on error.

OSSL_STORE_find_fn

This function takes a **OSSL_STORE_LOADER_CTX** pointer and a **OSSL_STORE_SEARCH** search criterion, and is used to tell the loader what to search for.

When called with the loader context being NULL, this function is expected to return 1 if the loader supports the criterion, otherwise 0.

When called with the loader context being something other than NULL, this function is expected to return 1 on success, 0 on error.

OSSL_STORE_load_fn

This function takes a **OSSL_STORE_LOADER_CTX** pointer and a **UI_METHOD** with associated data. It's expected to load the next available data, mold it into a data structure that can be wrapped in a **OSSL_STORE_INFO** using one of the **OSSL_STORE_INFO**(3) functions. If no more data is available or an error occurs, this function is expected to return NULL. The **OSSL_STORE_eof_fn** and **OSSL_STORE_error_fn** functions must indicate if it was in fact the end of data or if an error occurred.

Note that this function retrieves *one* data item only.

OSSL_STORE_eof_fn

This function takes a **OSSL_STORE_LOADER_CTX** pointer and is expected to return 1 to indicate that the end of available data has been reached. It is otherwise expected to return 0.

OSSL_STORE_error_fn

This function takes a **OSSL_STORE_LOADER_CTX** pointer and is expected to return 1 to indicate that an error occurred in a previous call to the **OSSL_STORE_load_fn** function. It is otherwise expected to return 0.

OSSL_STORE_close_fn
: This function takes a **OSSL_STORE_LOADER_CTX** pointer and is expected to close or shut down what needs to be closed, and finally free the contents of the **OSSL_STORE_LOADER_CTX** pointer. It returns 1 on success and 0 on error.

OSSL_STORE_LOADER_new() creates a new **OSSL_STORE_LOADER**. It takes an **ENGINE** *e* and a string *scheme*. *scheme* must *always* be set. Both *e* and *scheme* are used as is and must therefore be alive as long as the created loader is.

OSSL_STORE_LOADER_get0_engine() returns the engine of the *store_loader*.
OSSL_STORE_LOADER_get0_scheme() returns the scheme of the *store_loader*.

OSSL_STORE_LOADER_set_open() sets the opener function for the *store_loader*.

OSSL_STORE_LOADER_set_open_ex() sets the opener with library context function for the *store_loader*.

OSSL_STORE_LOADER_set_attach() sets the attacher function for the *store_loader*.

OSSL_STORE_LOADER_set_ctrl() sets the control function for the *store_loader*.

OSSL_STORE_LOADER_set_expect() sets the expect function for the *store_loader*.

OSSL_STORE_LOADER_set_load() sets the loader function for the *store_loader*.

OSSL_STORE_LOADER_set_eof() sets the end of file checker function for the *store_loader*.

OSSL_STORE_LOADER_set_close() sets the closing function for the *store_loader*.

OSSL_STORE_LOADER_free() frees the given *store_loader*.

OSSL_STORE_register_loader() register the given *store_loader* and thereby makes it available for use with **OSSL_STORE_open()**, **OSSL_STORE_open_ex()**, **OSSL_STORE_load()**, **OSSL_STORE_eof()** and **OSSL_STORE_close()**.

OSSL_STORE_unregister_loader() unregister the store loader for the given *scheme*.

RETURN VALUES

OSSL_STORE_LOADER_fetch() returns a pointer to an OSSL_STORE_LOADER object, or NULL on error.

OSSL_STORE_LOADER_up_ref() returns 1 on success, or 0 on error.

OSSL_STORE_LOADER_names_do_all() returns 1 if the callback was called for all names. A return value of 0 means that the callback was not called for any names.

OSSL_STORE_LOADER_free() doesn't return any value.

OSSL_STORE_LOADER_get0_provider() returns a pointer to a provider object, or NULL on error.

OSSL_STORE_LOADER_get0_properties() returns a pointer to a property definition string, or NULL on error.

OSSL_STORE_LOADER_is_a() returns 1 if *loader* was identifiable, otherwise 0.

OSSL_STORE_LOADER_get0_description() returns a pointer to a description, or NULL if there isn't one.

The functions with the types **OSSL_STORE_open_fn**, **OSSL_STORE_open_ex_fn**, **OSSL_STORE_ctrl_fn**, **OSSL_STORE_expect_fn**, **OSSL_STORE_load_fn**, **OSSL_STORE_eof_fn** and **OSSL_STORE_close_fn** have the same return values as **OSSL_STORE_open()**, **OSSL_STORE_open_ex()**, **OSSL_STORE_ctrl()**, **OSSL_STORE_expect()**, **OSSL_STORE_load()**, **OSSL_STORE_eof()** and **OSSL_STORE_close()**, respectively.

OSSL_STORE_LOADER_new() returns a pointer to a **OSSL_STORE_LOADER** on success, or NULL on failure.

OSSL_STORE_LOADER_set_open(), **OSSL_STORE_LOADER_set_open_ex()**, **OSSL_STORE_LOADER_set_ctrl()**, **OSSL_STORE_LOADER_set_load()**, **OSSL_STORE_LOADER_set_eof()** and **OSSL_STORE_LOADER_set_close()** return 1 on success, or 0 on failure.

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

OSSL_STORE_unregister_loader() returns the unregistered loader on success, or NULL on failure.

SEE ALSO

ossl_store(7), **OSSL_STORE_open**(3), **OSSL_LIB_CTX**(3), **provider-storemgmt**(7)

HISTORY

OSSL_STORE_LOADER_fetch(), **OSSL_STORE_LOADER_up_ref()**,

OSSL_STORE_LOADER_free(), **OSSL_STORE_LOADER_get0_provider()**, **OSSL_STORE_LOADER_get0_properties()**, **OSSL_STORE_LOADER_is_a()**, **OSSL_STORE_LOADER_do_all_provided()** and **OSSL_STORE_LOADER_names_do_all()** were added in OpenSSL 3.0.

OSSL_STORE_open_ex_fn() was added in OpenSSL 3.0.

OSSL_STORE_LOADER, **OSSL_STORE_LOADER_CTX**, **OSSL_STORE_LOADER_new()**, **OSSL_STORE_LOADER_set0_scheme()**, **OSSL_STORE_LOADER_get0_scheme()**, **OSSL_STORE_LOADER_get0_engine()**, **OSSL_STORE_LOADER_set_expect()**, **OSSL_STORE_LOADER_set_find()**, **OSSL_STORE_LOADER_set_attach()**, **OSSL_STORE_LOADER_set_open_ex()**, **OSSL_STORE_LOADER_set_open()**, **OSSL_STORE_LOADER_set_ctrl()**, **OSSL_STORE_LOADER_set_load()**, **OSSL_STORE_LOADER_set_eof()**, **OSSL_STORE_LOADER_set_close()**, **OSSL_STORE_LOADER_free()**, **OSSL_STORE_register_loader()**, **OSSL_STORE_LOADER_set_error()**, **OSSL_STORE_unregister_loader()**, **OSSL_STORE_open_fn()**, **OSSL_STORE_ctrl_fn()**, **OSSL_STORE_load_fn()**, **OSSL_STORE_eof_fn()** and **OSSL_STORE_close_fn()** were added in OpenSSL 1.1.1, and became deprecated in OpenSSL 3.0.

COPYRIGHT