NAME

X509_LOOKUP, X509_LOOKUP_TYPE, X509_LOOKUP_new, X509_LOOKUP_free, X509_LOOKUP_init, X509_LOOKUP_shutdown, X509_LOOKUP_set_method_data, X509_LOOKUP_get_method_data, X509_LOOKUP_ctrl_ex, X509_LOOKUP_ctrl, X509_LOOKUP_load_file_ex, X509_LOOKUP_load_file, X509_LOOKUP_add_dir, X509_LOOKUP_add_store_ex, X509_LOOKUP_add_store, X509_LOOKUP_load_store_ex, X509_LOOKUP_load_store, X509_LOOKUP_get_store, X509_LOOKUP_by_subject_ex, X509_LOOKUP_by_subject, X509_LOOKUP_by_issuer_serial, X509_LOOKUP_by_fingerprint, X509_LOOKUP_by_alias - OpenSSL certificate lookup mechanisms

SYNOPSIS

```
#include <openssl/x509_vfy.h>

typedef x509_lookup_st X509_LOOKUP;

typedef enum X509_LOOKUP_TYPE;

X509_LOOKUP *X509_LOOKUP_new(X509_LOOKUP_METHOD *method);
int X509_LOOKUP_init(X509_LOOKUP *ctx);
int X509_LOOKUP_shutdown(X509_LOOKUP *ctx);
void X509_LOOKUP_free(X509_LOOKUP *ctx);

int X509_LOOKUP_set_method_data(X509_LOOKUP *ctx, void *data);
void *X509_LOOKUP_get_method_data(const X509_LOOKUP *ctx);

int X509_LOOKUP_ctrl_ex(X509_LOOKUP *ctx, int cmd, const char *argc, long argl,
                        char **ret, OSSL_LIB_CTX *libctx, const char *propq);
int X509_LOOKUP_ctrl(X509_LOOKUP *ctx, int cmd, const char *argc,
                     long argl, char **ret);
int X509_LOOKUP_load_file_ex(X509_LOOKUP *ctx, char *name, long type,
                             OSSL_LIB_CTX *libctx, const char *propq);
int X509_LOOKUP_load_file(X509_LOOKUP *ctx, char *name, long type);
int X509_LOOKUP_load_file_ex(X509_LOOKUP *ctx, char *name, long type,
                             OSSL_LIB_CTX *libctx, const char *propq);
int X509_LOOKUP_add_dir(X509_LOOKUP *ctx, char *name, long type);
int X509_LOOKUP_add_store_ex(X509_LOOKUP *ctx, char *uri, OSSL_LIB_CTX *libctx,
                             const char *propq);
int X509_LOOKUP_add_store(X509_LOOKUP *ctx, char *uri);
int X509_LOOKUP_load_store_ex(X509_LOOKUP *ctx, char *uri, OSSL_LIB_CTX *libctx,
                              const char *propq);
```

```
int X509_LOOKUP_load_store(X509_LOOKUP *ctx, char *uri);

X509_STORE *X509_LOOKUP_get_store(const X509_LOOKUP *ctx);

int X509_LOOKUP_by_subject_ex(X509_LOOKUP *ctx, X509_LOOKUP_TYPE type,
                              const X509_NAME *name, X509_OBJECT *ret,
                              OSSL_LIB_CTX *libctx, const char *propq);
int X509_LOOKUP_by_subject(X509_LOOKUP *ctx, X509_LOOKUP_TYPE type,
                           const X509_NAME *name, X509_OBJECT *ret);
int X509_LOOKUP_by_issuer_serial(X509_LOOKUP *ctx, X509_LOOKUP_TYPE type,
                                 const X509_NAME *name,
                                 const ASN1_INTEGER *serial, X509_OBJECT *ret);
int X509_LOOKUP_by_fingerprint(X509_LOOKUP *ctx, X509_LOOKUP_TYPE type,
                               const unsigned char *bytes, int len,
                               X509_OBJECT *ret);
int X509_LOOKUP_by_alias(X509_LOOKUP *ctx, X509_LOOKUP_TYPE type,
                         const char *str, int len, X509_OBJECT *ret);
```

DESCRIPTION

The **X509_LOOKUP** structure holds the information needed to look up certificates and CRLs according to an associated **X509_LOOKUP_METHOD**(3). Multiple **X509_LOOKUP** instances can be added to an **X509_STORE**(3) to enable lookup in that store.

X509_LOOKUP_new() creates a new **X509_LOOKUP** using the given lookup *method*. It can also be created by calling **X509_STORE_add_lookup**(3), which will associate a **X509_STORE** with the lookup mechanism.

X509_LOOKUP_init() initializes the internal state and resources as needed by the given **X509_LOOKUP** to do its work.

X509_LOOKUP_shutdown() tears down the internal state and resources of the given **X509_LOOKUP**.

X509_LOOKUP_free() destructs the given **X509_LOOKUP**.

X509_LOOKUP_set_method_data() and **X509_LOOKUP_get_method_data()** associates and retrieves a pointer to application data to and from the given **X509_LOOKUP**, respectively.

X509_LOOKUP_ctrl_ex() is used to set or get additional data to or from a **X509_LOOKUP** structure or its associated **X509_LOOKUP_METHOD**(3). The arguments of the control command are passed via *argc* and *argl*, its return value via **ret*. The library context *libctx* and property query *propq* are used

when fetching algorithms from providers. The meaning of the arguments depends on the *cmd* number of the control command. In general, this function is not called directly, but wrapped by a macro call, see below. The control *cmd*s known to OpenSSL are discussed in more depth in "Control Commands".

X509_LOOKUP_ctrl() is similar to **X509_LOOKUP_ctrl_ex()** but uses NULL for the library context *libctx* and property query *propq*.

X509_LOOKUP_load_file_ex() passes a filename to be loaded immediately into the associated **X509_STORE**. The library context *libctx* and property query *propq* are used when fetching algorithms from providers. *type* indicates what type of object is expected. This can only be used with a lookup using the implementation **X509_LOOKUP_file**(3).

X509_LOOKUP_load_file() is similar to **X509_LOOKUP_load_file_ex()** but uses NULL for the library context *libctx* and property query *propq*.

X509_LOOKUP_add_dir() passes a directory specification from which certificates and CRLs are loaded on demand into the associated **X509_STORE**. *type* indicates what type of object is expected. This can only be used with a lookup using the implementation **X509_LOOKUP_hash_dir**(3).

X509_LOOKUP_add_store_ex() passes a URI for a directory-like structure from which containers with certificates and CRLs are loaded on demand into the associated **X509_STORE**. The library context *libctx* and property query *propq* are used when fetching algorithms from providers.

X509_LOOKUP_add_store() is similar to **X509_LOOKUP_add_store_ex()** but uses NULL for the library context *libctx* and property query *propq*.

X509_LOOKUP_load_store_ex() passes a URI for a single container from which certificates and CRLs are immediately loaded into the associated **X509_STORE**. The library context *libctx* and property query *propq* are used when fetching algorithms from providers. These functions can only be used with a lookup using the implementation **X509_LOOKUP_store**(3).

X509_LOOKUP_load_store() is similar to **X509_LOOKUP_load_store_ex()** but uses NULL for the library context *libctx* and property query *propq*.

X509_LOOKUP_load_file_ex(), **X509_LOOKUP_load_file()**, **X509_LOOKUP_add_dir()**, **X509_LOOKUP_add_store_ex()** **X509_LOOKUP_add_store()**, **X509_LOOKUP_load_store_ex()** and **X509_LOOKUP_load_store()** are implemented as macros that use **X509_LOOKUP_ctrl()**.

X509_LOOKUP_by_subject_ex(), **X509_LOOKUP_by_subject()**, **X509_LOOKUP_by_issuer_serial()**, **X509_LOOKUP_by_fingerprint()**, and

X509_LOOKUP_by_alias() look up certificates and CRLs in the **X509_STORE**(3) associated with the **X509_LOOKUP** using different criteria, where the looked up object is stored in *ret*. Some of the underlying **X509_LOOKUP_METHOD**s will also cache objects matching the criteria in the associated **X509_STORE**, which makes it possible to handle cases where the criteria have more than one hit.

Control Commands

The **X509_LOOKUP_METHOD**s built into OpenSSL recognize the following **X509_LOOKUP_ctrl()** *cmd*s:

X509_L_FILE_LOAD

This is the command that **X509_LOOKUP_load_file_ex()** and **X509_LOOKUP_load_file()** use. The filename is passed in *argc*, and the type in *argl*.

X509_L_ADD_DIR

This is the command that **X509_LOOKUP_add_dir()** uses. The directory specification is passed in *argc*, and the type in *argl*.

X509_L_ADD_STORE

This is the command that **X509_LOOKUP_add_store_ex()** and **X509_LOOKUP_add_store()** use. The URI is passed in *argc*.

X509_L_LOAD_STORE

This is the command that **X509_LOOKUP_load_store_ex()** and **X509_LOOKUP_load_store()** use. The URI is passed in *argc*.

RETURN VALUES

X509_LOOKUP_new() returns a **X509_LOOKUP** pointer when successful, or NULL on error.

X509_LOOKUP_init() and **X509_LOOKUP_shutdown()** return 1 on success, or 0 on error.

X509_LOOKUP_ctrl() returns -1 if the **X509_LOOKUP** doesn't have an associated **X509_LOOKUP_METHOD**, or 1 if the doesn't have a control function. Otherwise, it returns what the control function in the **X509_LOOKUP_METHOD** returns, which is usually 1 on success and 0 in error.

X509_LOOKUP_get_store() returns a **X509_STORE** pointer if there is one, otherwise NULL.

X509_LOOKUP_by_subject_ex(), **X509_LOOKUP_by_subject()**, **X509_LOOKUP_by_issuer_serial()**, **X509_LOOKUP_by_fingerprint()**, and **X509_LOOKUP_by_alias()** all return 0 if there is no **X509_LOOKUP_METHOD** or that method

doesn’t implement the corresponding function. Otherwise, it returns what the corresponding function in the **X509_LOOKUP_METHOD** returns, which is usually 1 on success and 0 in error.

SEE ALSO

X509_LOOKUP_METHOD(3), **X509_STORE**(3)

HISTORY

The functions **X509_LOOKUP_by_subject_ex()** and **X509_LOOKUP_ctrl_ex()** were added in OpenSSL 3.0.

The macros **X509_LOOKUP_load_file_ex()**, **X509_LOOKUP_load_store_ex()** and 509_LOOKUP_add_store_ex() were added in OpenSSL 3.0.

COPYRIGHT