Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'X509_STORE_set_purpose.3ossl' command

$ man X509_STORE_set_purpose.3ossl

X509_STORE_ADD_CERT(3ossl) OpenSSL X509_STORE_ADD_CERT(3ossl)

NAME

X509_STORE, X509_STORE_add_cert, X509_STORE_add_crl,

X509_STORE_set_depth, X509_STORE_set_flags, X509_STORE_set_purpose,

X509_STORE_set_trust, X509_STORE_add_lookup, X509_STORE_load_file_ex,

X509_STORE_load_file, X509_STORE_load_path, X509_STORE_load_store_ex,

X509_STORE_load_store, X509_STORE_set_default_paths_ex,

X509_STORE_set_default_paths, X509_STORE_load_locations_ex,

X509_STORE_load_locations - X509_STORE manipulation

SYNOPSIS

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

typedef x509_store_st X509_STORE;

int X509_STORE_add_cert(X509_STORE *ctx, X509 *x);

int X509_STORE_add_crl(X509_STORE *ctx, X509_CRL *x);

int X509_STORE_set_depth(X509_STORE *store, int depth);

int X509_STORE_set_flags(X509_STORE *ctx, unsigned long flags);

int X509_STORE_set_purpose(X509_STORE *ctx, int purpose);

int X509_STORE_set_trust(X509_STORE *ctx, int trust);

X509_LOOKUP *X509_STORE_add_lookup(X509_STORE *store,
                                   X509_LOOKUP_METHOD *meth);

int X509_STORE_set_default_paths_ex(X509_STORE *ctx, OSSL_LIB_CTX *libctx,
                                    const char *propq);

int X509_STORE_set_default_paths(X509_STORE *ctx);
```

```
int X509_STORE_load_file_ex(X509_STORE *ctx, const char *file,
                            OSSL_LIB_CTX *libctx, const char *propq);
int X509_STORE_load_file(X509_STORE *ctx, const char *file);
int X509_STORE_load_path(X509_STORE *ctx, const char *dir);
int X509_STORE_load_store_ex(X509_STORE *ctx, const char *uri,
                             OSSL_LIB_CTX *libctx, const char *propq);
int X509_STORE_load_store(X509_STORE *ctx, const char *uri);
int X509_STORE_load_locations_ex(X509_STORE *ctx, const char *file,
                                 const char *dir, OSSL_LIB_CTX *libctx,
                                 const char *propq);
int X509_STORE_load_locations(X509_STORE *ctx,
                              const char *file, const char *dir);
```

DESCRIPTION

The X509_STORE structure is intended to be a consolidated mechanism for holding information about X.509 certificates and CRLs, and constructing and validating chains of certificates terminating in trusted roots. It admits multiple lookup mechanisms and efficient scaling performance with large numbers of certificates, and a great deal of flexibility in how validation and policy checks are performed.

Details of the chain building and checking process are described in "Certification Path Building" in openssl-verification-options(1) and "Certification Path Validation" in openssl-verification-options(1).

X509_STORE_new(3) creates an empty X509_STORE structure, which contains no information about trusted certificates or where such certificates are located on disk, and is generally not usable. Normally, trusted certificates will be added to the X509_STORE to prepare it for use, via mechanisms such as X509_STORE_add_lookup() and X509_LOOKUP_file(), or PEM_read_bio_X509_AUX() and X509_STORE_add_cert(). CRLs can also be added, and many behaviors configured as desired.

Once the X509_STORE is suitably configured, X509_STORE_CTX_new() is used to instantiate a single-use X509_STORE_CTX for each chain-building and verification operation. That process includes providing the end-entity certificate to be verified and an additional set of untrusted

certificates that may be used in chain-building. As such, it is expected that the certificates included in the X509_STORE are certificates that represent trusted entities such as root certificate authorities (CAs). OpenSSL represents these trusted certificates internally as X509 objects with an associated X509_CERT_AUX, as are produced by PEM_read_bio_X509_AUX() and similar routines that refer to X509_AUX. The public interfaces that operate on such trusted certificates still operate on pointers to X509 objects, though.

X509_STORE_add_cert() and X509_STORE_add_crl() add the respective object to the X509_STORE's local storage. Untrusted objects should not be added in this way. The added object's reference count is incremented by one, hence the caller retains ownership of the object and needs to free it when it is no longer needed.

X509_STORE_set_depth(), X509_STORE_set_flags(), X509_STORE_set_purpose(), X509_STORE_set_trust(), and X509_STORE_set1_param() set the default values for the corresponding values used in certificate chain validation. Their behavior is documented in the corresponding X509_VERIFY_PARAM manual pages, e.g., X509_VERIFY_PARAM_set_depth(3).

X509_STORE_add_lookup() finds or creates a X509_LOOKUP(3) with the X509_LOOKUP_METHOD(3) meth and adds it to the X509_STORE store. This also associates the X509_STORE with the lookup, so X509_LOOKUP functions can look up objects in that store.

X509_STORE_load_file_ex() loads trusted certificate(s) into an X509_STORE from a given file. The library context libctx and property query propq are used when fetching algorithms from providers.

X509_STORE_load_file() is similar to X509_STORE_load_file_ex() but uses NULL for the library context libctx and property query propq.

X509_STORE_load_path() loads trusted certificate(s) into an X509_STORE from a given directory path. The certificates in the directory must be in hashed form, as documented in X509_LOOKUP_hash_dir(3).

X509_STORE_load_store_ex() loads trusted certificate(s) into an X509_STORE from a store at a given URI. The library context libctx and

property query propq are used when fetching algorithms from providers.

X509_STORE_load_store() is similar to X509_STORE_load_store_ex() but uses NULL for the library context libctx and property query propq.

X509_STORE_load_locations_ex() combines X509_STORE_load_file_ex() and X509_STORE_load_path() for a given file and/or directory path. It is permitted to specify just a file, just a directory, or both paths.

X509_STORE_load_locations() is similar to X509_STORE_load_locations_ex() but uses NULL for the library context libctx and property query propq.

X509_STORE_set_default_paths_ex() is somewhat misnamed, in that it does not set what default paths should be used for loading certificates. Instead, it loads certificates into the X509_STORE from the hardcoded default paths. The library context libctx and property query propq are used when fetching algorithms from providers.

X509_STORE_set_default_paths() is similar to X509_STORE_set_default_paths_ex() but uses NULL for the library context libctx and property query propq.

RETURN VALUES

X509_STORE_add_cert(), X509_STORE_add_crl(), X509_STORE_set_depth(), X509_STORE_set_flags(), X509_STORE_set_purpose(), X509_STORE_set_trust(), X509_STORE_load_file_ex(), X509_STORE_load_file(), X509_STORE_load_path(), X509_STORE_load_store_ex(), X509_STORE_load_store(), X509_STORE_load_locations_ex(), X509_STORE_load_locations(), X509_STORE_set_default_paths_ex() and X509_STORE_set_default_paths() return 1 on success or 0 on failure.

X509_STORE_add_lookup() returns the found or created X509_LOOKUP(3), or NULL on error.

SEE ALSO

X509_LOOKUP_hash_dir(3). X509_VERIFY_PARAM_set_depth(3). X509_STORE_new(3), X509_STORE_get0_param(3)

HISTORY

The functions X509_STORE_set_default_paths_ex(),

X509_STORE_load_file_ex(), X509_STORE_load_store_ex() and X509_STORE_load_locations_ex() were added in OpenSSL 3.0.

COPYRIGHT