NAME

X509_STORE_CTX_get_error, X509_STORE_CTX_set_error, X509_STORE_CTX_get_error_depth, X509_STORE_CTX_set_error_depth, X509_STORE_CTX_get_current_cert, X509_STORE_CTX_set_current_cert, X509_STORE_CTX_get0_cert, X509_STORE_CTX_get1_chain, X509_verify_cert_error_string - get or set certificate verification status information

SYNOPSIS

```
#include <openssl/x509.h>

int   X509_STORE_CTX_get_error(const X509_STORE_CTX *ctx);
void  X509_STORE_CTX_set_error(X509_STORE_CTX *ctx, int s);
int   X509_STORE_CTX_get_error_depth(const X509_STORE_CTX *ctx);
void  X509_STORE_CTX_set_error_depth(X509_STORE_CTX *ctx, int depth);
X509 *X509_STORE_CTX_get_current_cert(const X509_STORE_CTX *ctx);
void  X509_STORE_CTX_set_current_cert(X509_STORE_CTX *ctx, X509 *x);
X509 *X509_STORE_CTX_get0_cert(const X509_STORE_CTX *ctx);

STACK_OF(X509) *X509_STORE_CTX_get1_chain(const X509_STORE_CTX *ctx);

const char *X509_verify_cert_error_string(long n);
```

DESCRIPTION

These functions are typically called after certificate or chain verification using **X509_verify_cert**(3) or **X509_STORE_CTX_verify**(3) has indicated an error or in a verification callback to determine the nature of an error.

X509_STORE_CTX_get_error() returns the error code of *ctx*. See the "ERROR CODES" section for a full description of all error codes. It may return a code != X509_V_OK even if **X509_verify_cert()** did not indicate an error, likely because a verification callback function has waived the error.

X509_STORE_CTX_set_error() sets the error code of *ctx* to *s*. For example it might be used in a verification callback to set an error based on additional checks.

X509_STORE_CTX_get_error_depth() returns the *depth* of the error. This is a nonnegative integer representing where in the certificate chain the error occurred. If it is zero it occurred in the end entity certificate, one if it is the certificate which signed the end entity certificate and so on.

X509_STORE_CTX_set_error_depth() sets the error *depth*. This can be used in combination with **X509_STORE_CTX_set_error()** to set the depth at which an error condition was detected.

X509_STORE_CTX_get_current_cert() returns the current certificate in *ctx*. If an error occurred, the current certificate will be the one that is most closely related to the error, or possibly NULL if no such certificate is relevant.

X509_STORE_CTX_set_current_cert() sets the certificate *x* in *ctx* which caused the error. This value is not intended to remain valid for very long, and remains owned by the caller. It may be examined by a verification callback invoked to handle each error encountered during chain verification and is no longer required after such a callback. If a callback wishes the save the certificate for use after it returns, it needs to increment its reference count via **X509_up_ref**(3). Once such a *saved* certificate is no longer needed it can be freed with **X509_free**(3).

X509_STORE_CTX_get0_cert() retrieves an internal pointer to the certificate being verified by the *ctx*.

X509_STORE_CTX_get1_chain() returns a complete validate chain if a previous verification is successful. Otherwise the returned chain may be incomplete or invalid. The returned chain persists after the *ctx* structure is freed. When it is no longer needed it should be free up using:

```
sk_X509_pop_free(chain, X509_free);
```

X509_verify_cert_error_string() returns a human readable error string for verification error *n*.

RETURN VALUES

X509_STORE_CTX_get_error() returns **X509_V_OK** or an error code.

X509_STORE_CTX_get_error_depth() returns a nonnegative error depth.

X509_STORE_CTX_get_current_cert() returns the certificate which caused the error or NULL if no certificate is relevant to the error.

X509_verify_cert_error_string() returns a human readable error string for verification error *n*.

ERROR CODES

A list of error codes and messages is shown below. Some of the error codes are defined but currently never returned: these are described as "unused".

X509_V_OK: ok

The operation was successful.

X509_V_ERR_UNSPECIFIED: unspecified certificate verification error

Unspecified error; should not happen.

X509_V_ERR_UNABLE_TO_GET_ISSUER_CERT: unable to get issuer certificate
The issuer certificate of a locally looked up certificate could not be found. This normally means the list of trusted certificates is not complete. To allow any certificate (not only a self-signed one) in the trust store to terminate the chain the **X509_V_FLAG_PARTIAL_CHAIN** flag may be set.

X509_V_ERR_UNABLE_TO_GET_CRL: unable to get certificate CRL
The CRL of a certificate could not be found.

X509_V_ERR_UNABLE_TO_DECRYPT_CERT_SIGNATURE: unable to decrypt certificate's signature
The certificate signature could not be decrypted. This means that the actual signature value could not be determined rather than it not matching the expected value, this is only meaningful for RSA keys.

X509_V_ERR_UNABLE_TO_DECRYPT_CRL_SIGNATURE: unable to decrypt CRL's signature
The CRL signature could not be decrypted: this means that the actual signature value could not be determined rather than it not matching the expected value. Unused.

X509_V_ERR_UNABLE_TO_DECODE_ISSUER_PUBLIC_KEY: unable to decode issuer public key The public key in the certificate "SubjectPublicKeyInfo" field could not be read.

X509_V_ERR_CERT_SIGNATURE_FAILURE: certificate signature failure
The signature of the certificate is invalid.

X509_V_ERR_CRL_SIGNATURE_FAILURE: CRL signature failure
The signature of the CRL is invalid.

X509_V_ERR_CERT_NOT_YET_VALID: certificate is not yet valid
The certificate is not yet valid: the "notBefore" date is after the current time.

X509_V_ERR_CERT_HAS_EXPIRED: certificate has expired
The certificate has expired: that is the "notAfter" date is before the current time.

X509_V_ERR_CRL_NOT_YET_VALID: CRL is not yet valid
The CRL is not yet valid.

X509_V_ERR_CRL_HAS_EXPIRED: CRL has expired
The CRL has expired.

X509_V_ERR_ERROR_IN_CERT_NOT_BEFORE_FIELD: format error in certificate's notBefore field

The certificate "notBefore" field contains an invalid time.

X509_V_ERR_ERROR_IN_CERT_NOT_AFTER_FIELD: format error in certificate's notAfter field

The certificate "notAfter" field contains an invalid time.

X509_V_ERR_ERROR_IN_CRL_LAST_UPDATE_FIELD: format error in CRL's lastUpdate field

The CRL **lastUpdate** field contains an invalid time.

X509_V_ERR_ERROR_IN_CRL_NEXT_UPDATE_FIELD: format error in CRL's nextUpdate field

The CRL "nextUpdate" field contains an invalid time.

X509_V_ERR_OUT_OF_MEM: out of memory

An error occurred trying to allocate memory.

X509_V_ERR_DEPTH_ZERO_SELF_SIGNED_CERT: self-signed certificate

The passed certificate is self-signed and the same certificate cannot be found in the list of trusted certificates.

X509_V_ERR_SELF_SIGNED_CERT_IN_CHAIN: self-signed certificate in certificate chain

The certificate chain could be built up using the untrusted certificates but no suitable trust anchor (which typically is a self-signed root certificate) could be found in the trust store.

X509_V_ERR_UNABLE_TO_GET_ISSUER_CERT_LOCALLY: unable to get local issuer certificate

The issuer certificate could not be found: this occurs if the issuer certificate of an untrusted certificate cannot be found.

X509_V_ERR_UNABLE_TO_VERIFY_LEAF_SIGNATURE: unable to verify the first certificate

No signatures could be verified because the chain contains only one certificate and it is not self-signed and the **X509_V_FLAG_PARTIAL_CHAIN** flag is not set.

X509_V_ERR_CERT_CHAIN_TOO_LONG: certificate chain too long

The certificate chain length is greater than the supplied maximum depth.

X509_V_ERR_CERT_REVOKED: certificate revoked

The certificate has been revoked.

X509_V_ERR_NO_ISSUER_PUBLIC_KEY: issuer certificate doesn't have a public key

The issuer certificate does not have a public key.

X509_V_ERR_PATH_LENGTH_EXCEEDED: path length constraint exceeded
The basicConstraints path-length parameter has been exceeded.

X509_V_ERR_INVALID_PURPOSE: unsuitable certificate purpose
The target certificate cannot be used for the specified purpose.

X509_V_ERR_CERT_UNTRUSTED: certificate not trusted
The root CA is not marked as trusted for the specified purpose.

X509_V_ERR_CERT_REJECTED: certificate rejected
The root CA is marked to reject the specified purpose.

X509_V_ERR_SUBJECT_ISSUER_MISMATCH: subject issuer mismatch
The current candidate issuer certificate was rejected because its subject name did not match the issuer name of the current certificate.

X509_V_ERR_AKID_SKID_MISMATCH: authority and subject key identifier mismatch
The current candidate issuer certificate was rejected because its subject key identifier was present and did not match the authority key identifier current certificate.

X509_V_ERR_AKID_ISSUER_SERIAL_MISMATCH: authority and issuer serial number mismatch
The current candidate issuer certificate was rejected because its issuer name and serial number was present and did not match the authority key identifier of the current certificate.

X509_V_ERR_KEYUSAGE_NO_CERTSIGN: key usage does not include certificate signing
The current candidate issuer certificate was rejected because its "keyUsage" extension does not permit certificate signing.

X509_V_ERR_UNABLE_TO_GET_CRL_ISSUER: unable to get CRL issuer certificate
Unable to get CRL issuer certificate.

X509_V_ERR_UNHANDLED_CRITICAL_EXTENSION: unhandled critical extension
Unhandled critical extension.

X509_V_ERR_KEYUSAGE_NO_CRL_SIGN: key usage does not include CRL signing
Key usage does not include CRL signing.

X509_V_ERR_UNHANDLED_CRITICAL_CRL_EXTENSION: unhandled critical CRL extension

Unhandled critical CRL extension.

X509_V_ERR_INVALID_NON_CA: invalid non-CA certificate (has CA markings)
Invalid non-CA certificate has CA markings.

X509_V_ERR_PROXY_PATH_LENGTH_EXCEEDED: proxy path length constraint exceeded
Proxy path length constraint exceeded.

X509_V_ERR_KEYUSAGE_NO_DIGITAL_SIGNATURE: key usage does not include digital signature
Key usage does not include digital signature, and therefore cannot sign certificates.

X509_V_ERR_PROXY_CERTIFICATES_NOT_ALLOWED: proxy certificates not allowed, please set the appropriate flag
Proxy certificates not allowed unless the **X509_V_FLAG_ALLOW_PROXY_CERTS** flag is set.

X509_V_ERR_INVALID_EXTENSION: invalid or inconsistent certificate extension
A certificate extension had an invalid value (for example an incorrect encoding) or some value inconsistent with other extensions.

X509_V_ERR_INVALID_POLICY_EXTENSION: invalid or inconsistent certificate policy extension
A certificate policies extension had an invalid value (for example an incorrect encoding) or some value inconsistent with other extensions. This error only occurs if policy processing is enabled.

X509_V_ERR_NO_EXPLICIT_POLICY: no explicit policy
The verification flags were set to require and explicit policy but none was present.

X509_V_ERR_DIFFERENT_CRL_SCOPE: different CRL scope
The only CRLs that could be found did not match the scope of the certificate.

X509_V_ERR_UNSUPPORTED_EXTENSION_FEATURE: unsupported extension feature
Some feature of a certificate extension is not supported. Unused.

X509_V_ERR_UNNESTED_RESOURCE: RFC 3779 resource not subset of parent's resources
See RFC 3779 for details.

X509_V_ERR_PERMITTED_VIOLATION: permitted subtree violation
A name constraint violation occurred in the permitted subtrees.

X509_V_ERR_EXCLUDED_VIOLATION: excluded subtree violation

A name constraint violation occurred in the excluded subtrees.

X509_V_ERR_SUBTREE_MINMAX: name constraints minimum and maximum not supported
A certificate name constraints extension included a minimum or maximum field: this is not supported.

X509_V_ERR_APPLICATION_VERIFICATION: application verification failure
An application specific error. This will never be returned unless explicitly set by an application callback.

X509_V_ERR_UNSUPPORTED_CONSTRAINT_TYPE: unsupported name constraint type
An unsupported name constraint type was encountered. OpenSSL currently only supports directory name, DNS name, email and URI types.

X509_V_ERR_UNSUPPORTED_CONSTRAINT_SYNTAX: unsupported or invalid name constraint syntax
The format of the name constraint is not recognised: for example an email address format of a form not mentioned in RFC3280. This could be caused by a garbage extension or some new feature not currently supported.

X509_V_ERR_UNSUPPORTED_NAME_SYNTAX: unsupported or invalid name syntax
Unsupported or invalid name syntax.

X509_V_ERR_CRL_PATH_VALIDATION_ERROR: CRL path validation error
An error occurred when attempting to verify the CRL path. This error can only happen if extended CRL checking is enabled.

X509_V_ERR_PATH_LOOP: path loop
Path loop.

X509_V_ERR_HOSTNAME_MISMATCH: hostname mismatch
Hostname mismatch.

X509_V_ERR_EMAIL_MISMATCH: email address mismatch
Email address mismatch.

X509_V_ERR_IP_ADDRESS_MISMATCH: IP address mismatch
IP address mismatch.

X509_V_ERR_DANE_NO_MATCH: no matching DANE TLSA records

DANE TLSA authentication is enabled, but no TLSA records matched the certificate chain. This error is only possible in **openssl-s_client**(1).

X509_V_ERR_EE_KEY_TOO_SMALL: EE certificate key too weak
EE certificate key too weak.

X509_V_ERR_CA_KEY_TOO_SMALL: CA certificate key too weak
CA certificate key too weak.

X509_V_ERR_CA_MD_TOO_WEAK: CA signature digest algorithm too weak
CA signature digest algorithm too weak.

X509_V_ERR_INVALID_CALL: invalid certificate verification context
Invalid certificate verification context.

X509_V_ERR_STORE_LOOKUP: issuer certificate lookup error
Issuer certificate lookup error.

X509_V_ERR_NO_VALID_SCTS: certificate transparency required, but no valid SCTs found
Certificate Transparency required, but no valid SCTs found.

X509_V_ERR_PROXY_SUBJECT_NAME_VIOLATION: proxy subject name violation
Proxy subject name violation.

X509_V_ERR_OCSP_VERIFY_NEEDED: OCSP verification needed
Returned by the verify callback to indicate an OCSP verification is needed.

X509_V_ERR_OCSP_VERIFY_FAILED: OCSP verification failed
Returned by the verify callback to indicate OCSP verification failed.

X509_V_ERR_OCSP_CERT_UNKNOWN: OCSP unknown cert
Returned by the verify callback to indicate that the certificate is not recognized by the OCSP responder.

X509_V_ERR_UNSUPPORTED_SIGNATURE_ALGORITHM: unsupported signature algorithm
Cannot find certificate signature algorithm.

X509_V_ERR_SIGNATURE_ALGORITHM_MISMATCH: subject signature algorithm and issuer public key algorithm mismatch
The issuer’s public key is not of the type required by the signature in the subject’s certificate.

X509_V_ERR_SIGNATURE_ALGORITHM_INCONSISTENCY: cert info signature and signature algorithm mismatch

The algorithm given in the certificate info is inconsistent with the one used for the certificate signature.

X509_V_ERR_INVALID_CA: invalid CA certificate

A CA certificate is invalid. Either it is not a CA or its extensions are not consistent with the supplied purpose.

NOTES

The above functions should be used instead of directly referencing the fields in the **X509_VERIFY_CTX** structure.

In versions of OpenSSL before 1.0 the current certificate returned by **X509_STORE_CTX_get_current_cert()** was never NULL. Applications should check the return value before printing out any debugging information relating to the current certificate.

If an unrecognised error code is passed to **X509_verify_cert_error_string()** the numerical value of the unknown code is returned in a static buffer. This is not thread safe but will never happen unless an invalid code is passed.

BUGS

Previous versions of this documentation swapped the meaning of the **X509_V_ERR_UNABLE_TO_GET_ISSUER_CERT** and **X509_V_ERR_UNABLE_TO_GET_ISSUER_CERT_LOCALLY** error codes.

SEE ALSO

X509_verify_cert(3), **X509_STORE_CTX_verify**(3), **X509_up_ref**(3), **X509_free**(3).

COPYRIGHT