NAME

SSL_CTX_set0_chain, SSL_CTX_set1_chain, SSL_CTX_add0_chain_cert, SSL_CTX_add1_chain_cert, SSL_CTX_get0_chain_certs, SSL_CTX_clear_chain_certs, SSL_set0_chain, SSL_set1_chain, SSL_add0_chain_cert, SSL_add1_chain_cert, SSL_get0_chain_certs, SSL_clear_chain_certs, SSL_CTX_build_cert_chain, SSL_build_cert_chain, SSL_CTX_select_current_cert, SSL_select_current_cert, SSL_CTX_set_current_cert, SSL_set_current_cert - extra chain certificate processing

SYNOPSIS

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

int SSL_CTX_set0_chain(SSL_CTX *ctx, STACK_OF(X509) *sk);
int SSL_CTX_set1_chain(SSL_CTX *ctx, STACK_OF(X509) *sk);
int SSL_CTX_add0_chain_cert(SSL_CTX *ctx, X509 *x509);
int SSL_CTX_add1_chain_cert(SSL_CTX *ctx, X509 *x509);
int SSL_CTX_get0_chain_certs(SSL_CTX *ctx, STACK_OF(X509) **sk);
int SSL_CTX_clear_chain_certs(SSL_CTX *ctx);

int SSL_set0_chain(SSL *ssl, STACK_OF(X509) *sk);
int SSL_set1_chain(SSL *ssl, STACK_OF(X509) *sk);
int SSL_add0_chain_cert(SSL *ssl, X509 *x509);
int SSL_add1_chain_cert(SSL *ssl, X509 *x509);
int SSL_get0_chain_certs(SSL *ssl, STACK_OF(X509) **sk);
int SSL_clear_chain_certs(SSL *ssl);

int SSL_CTX_build_cert_chain(SSL_CTX *ctx, flags);
int SSL_build_cert_chain(SSL *ssl, flags);

int SSL_CTX_select_current_cert(SSL_CTX *ctx, X509 *x509);
int SSL_select_current_cert(SSL *ssl, X509 *x509);
int SSL_CTX_set_current_cert(SSL_CTX *ctx, long op);
int SSL_set_current_cert(SSL *ssl, long op);
```

DESCRIPTION

SSL_CTX_set0_chain() and *SSL_CTX_set1_chain()* set the certificate chain associated with the current certificate of **ctx** to **sk**.

SSL_CTX_add0_chain_cert() and *SSL_CTX_add1_chain_cert()* append the single certificate **x509** to the chain associated with the current certificate of **ctx**.

SSL_CTX_get0_chain_certs() retrieves the chain associated with the current certificate of **ctx**.

SSL_CTX_clear_chain_certs() clears any existing chain associated with the current certificate of **ctx**. (This is implemented by calling *SSL_CTX_set0_chain()* with **sk** set to **NULL**).

SSL_CTX_build_cert_chain() builds the certificate chain for **ctx** normally this uses the chain store or the verify store if the chain store is not set. If the function is successful the built chain will replace any existing chain. The**flags** parameter can be set to **SSL_B UILD_CHAIN_FLAG_UNTRUSTED** to use existing chain certificates as untrusted CAs, **SSL_BUILD_CHAIN_FLAG_NO_ROOT** to omit the root CA from the built chain, **SSL_BUILD_CHAIN_FLAG_CHECK** to use all existing chain certificates only to build the chain (effectively sanity checking and rearranging them if necessary), the flag **SSL_BUILD_CHAIN_FLAG_IGNORE_ERROR** ignores any errors during verification: if flag **SSL_BUILD_CHAIN_FLAG_CLEAR_ERROR** is also set verification errors are cleared from the error queue.

Each of these functions operates on the *current* end entity (i.e. server or client) certificate. This is the last certificate loaded or selected on the corresponding **ctx** structure.

SSL_CTX_select_current_cert() selects **x509** as the current end entity certificate, but only if **x509** has

already been loaded into **ctx** using a function such as *SSL_CTX_use_certificate()*.

SSL_set0_chain(), *SSL_set1_chain()*, *SSL_add0_chain_cert()*, *SSL_add1_chain_cert()*, *SSL_get0_chain_certs()*, *SSL_clear_chain_certs()*, *SSL_build_cert_chain()*, *SSL_select_current_cert()* and *SSL_set_current_cert()* are similar except they apply to SSL structure **ssl**.

SSL_CTX_set_current_cert() changes the current certificate to a value based on the **op** argument. Currently **op** can be **SSL_CERT_SET_FIRST** to use the first valid certificate or **SSL_CERT_SET_NEXT** to set the next valid certificate after the current certificate. These two operations can be used to iterate over all certificates in an **SSL_CTX** structure.

SSL_set_current_cert() also supports the option **SSL_CERT_SET_SERVER**. If**ssl** is a serv er and has sent a certificate to a connected client this option sets that certificate to the current certificate and returns 1. If the negotiated ciphersuite is anonymous (and thus no certificate will be sent) 2 is returned and the current certificate is unchanged. If **ssl** is not a server or a certificate has not been sent 0 is returned and the current certificate is unchanged.

All these functions are implemented as macros. Those containing a **1** increment the reference count of the supplied certificate or chain so it must be freed at some point after the operation. Those containing a **0** do not increment reference counts and the supplied certificate or chain **MUST NOT** be freed after the operation.

NOTES

The chains associate with an SSL_CTX structure are copied to any SSL structures when *SSL_new()* is called. SSL structures will not be affected by any chains subsequently changed in the parent SSL_CTX.

One chain can be set for each key type supported by a server. So, for example, an RSA and a DSA certificate can (and often will) have different chains.

The functions *SSL_CTX_build_cert_chain()* and *SSL_build_cert_chain()* can be used to check application configuration and to ensure any necessary subordinate CAs are sent in the correct order. Misconfigured applications sending incorrect certificate chains often cause problems with peers.

For example an application can add any set of certificates using *SSL_CTX_use_certificate_chain_file()* then call *SSL_CTX_build_cert_chain()* with the option **SSL_BUILD_CHAIN_FLAG_CHECK** to check and reorder them.

Applications can issue non fatal warnings when checking chains by setting the flag **SSL_BUILD_CHAIN_FLAG_IGNORE_ERRORS** and checking the return value.

Calling *SSL_CTX_build_cert_chain()* or *SSL_build_cert_chain()* is more efficient than the automatic chain building as it is only performed once. Automatic chain building is performed on each new session.

If any certificates are added using these functions no certificates added using *SSL_CTX_add_extra_chain_cert()* will be used.

RETURN VALUES

SSL_set_current_cert() with **SSL_CERT_SET_SERVER** return 1 for success, 2 if no server certificate is used because the ciphersuites is anonymous and 0 for failure.

SSL_CTX_build_cert_chain() and *SSL_build_cert_chain()* return 1 for success and 0 for failure. If the flag **SSL_BUILD_CHAIN_FLAG_IGNORE_ERROR** and a verification error occurs then 2 is returned.

All other functions return 1 for success and 0 for failure.

SEE ALSO

SSL_CTX_add_extra_chain_cert(3)

HISTORY

These functions were first added to OpenSSL 1.0.2.

COPYRIGHT