NAME

SSL_CTX_set_client_CA_list, SSL_set_client_CA_list, SSL_get_client_CA_list, SSL_CTX_get_client_CA_list, SSL_CTX_add_client_CA, SSL_add_client_CA, SSL_set0_CA_list, SSL_CTX_set0_CA_list, SSL_get0_CA_list, SSL_CTX_get0_CA_list, SSL_add1_to_CA_list, SSL_CTX_add1_to_CA_list, SSL_get0_peer_CA_list - get or set CA list

SYNOPSIS

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

void SSL_CTX_set_client_CA_list(SSL_CTX *ctx, STACK_OF(X509_NAME) *list);
void SSL_set_client_CA_list(SSL *s, STACK_OF(X509_NAME) *list);
STACK_OF(X509_NAME) *SSL_get_client_CA_list(const SSL *s);
STACK_OF(X509_NAME) *SSL_CTX_get_client_CA_list(const SSL_CTX *ctx);
int SSL_CTX_add_client_CA(SSL_CTX *ctx, X509 *cacert);
int SSL_add_client_CA(SSL *ssl, X509 *cacert);

void SSL_CTX_set0_CA_list(SSL_CTX *ctx, STACK_OF(X509_NAME) *name_list);
void SSL_set0_CA_list(SSL *s, STACK_OF(X509_NAME) *name_list);
const STACK_OF(X509_NAME) *SSL_CTX_get0_CA_list(const SSL_CTX *ctx);
const STACK_OF(X509_NAME) *SSL_get0_CA_list(const SSL *s);
int SSL_CTX_add1_to_CA_list(SSL_CTX *ctx, const X509 *x);
int SSL_add1_to_CA_list(SSL *ssl, const X509 *x);

const STACK_OF(X509_NAME) *SSL_get0_peer_CA_list(const SSL *s);
```

DESCRIPTION

The functions described here set and manage the list of CA names that are sent between two communicating peers.

For TLS versions 1.2 and earlier the list of CA names is only sent from the server to the client when requesting a client certificate. So any list of CA names set is never sent from client to server and the list of CA names retrieved by **SSL_get0_peer_CA_list()** is always **NULL**.

For TLS 1.3 the list of CA names is sent using the **certificate_authorities** extension and may be sent by a client (in the ClientHello message) or by a server (when requesting a certificate).

In most cases it is not necessary to set CA names on the client side. The list of CA names that are acceptable to the client will be sent in plaintext to the server. This has privacy implications and may also have performance implications if the list is large. This optional capability was introduced as part of TLSv1.3 and therefore setting CA names on the client side will have no impact if that protocol version

has been disabled. Most servers do not need this and so this should be avoided unless required.

The "client CA list" functions below only have an effect when called on the server side.

SSL_CTX_set_client_CA_list() sets the **list** of CAs sent to the client when requesting a client certificate for **ctx**. Ownership of **list** is transferred to **ctx** and it should not be freed by the caller.

SSL_set_client_CA_list() sets the **list** of CAs sent to the client when requesting a client certificate for the chosen **ssl**, overriding the setting valid for **ssl**'s SSL_CTX object. Ownership of **list** is transferred to **s** and it should not be freed by the caller.

SSL_CTX_get_client_CA_list() returns the list of client CAs explicitly set for **ctx** using **SSL_CTX_set_client_CA_list()**. The returned list should not be freed by the caller.

SSL_get_client_CA_list() returns the list of client CAs explicitly set for **ssl** using **SSL_set_client_CA_list()** or **ssl**'s SSL_CTX object with **SSL_CTX_set_client_CA_list()**, when in server mode. In client mode, SSL_get_client_CA_list returns the list of client CAs sent from the server, if any. The returned list should not be freed by the caller.

SSL_CTX_add_client_CA() adds the CA name extracted from **cacert** to the list of CAs sent to the client when requesting a client certificate for **ctx**.

SSL_add_client_CA() adds the CA name extracted from **cacert** to the list of CAs sent to the client when requesting a client certificate for the chosen **ssl**, overriding the setting valid for **ssl**'s SSL_CTX object.

SSL_get0_peer_CA_list() retrieves the list of CA names (if any) the peer has sent. This can be called on either the server or the client side. The returned list should not be freed by the caller.

The "generic CA list" functions below are very similar to the "client CA list" functions except that they have an effect on both the server and client sides. The lists of CA names managed are separate - so you cannot (for example) set CA names using the "client CA list" functions and then get them using the "generic CA list" functions. Where a mix of the two types of functions has been used on the server side then the "client CA list" functions take precedence. Typically, on the server side, the "client CA list " functions should be used in preference. As noted above in most cases it is not necessary to set CA names on the client side.

SSL_CTX_set0_CA_list() sets the list of CAs to be sent to the peer to **name_list**. Ownership of **name_list** is transferred to **ctx** and it should not be freed by the caller.

SSL_set0_CA_list() sets the list of CAs to be sent to the peer to **name_list** overriding any list set in the parent **SSL_CTX** of **s**. Ownership of **name_list** is transferred to **s** and it should not be freed by the caller.

SSL_CTX_get0_CA_list() retrieves any previously set list of CAs set for **ctx**. The returned list should not be freed by the caller.

SSL_get0_CA_list() retrieves any previously set list of CAs set for **s** or if none are set the list from the parent **SSL_CTX** is retrieved. The returned list should not be freed by the caller.

SSL_CTX_add1_to_CA_list() appends the CA subject name extracted from **x** to the list of CAs sent to peer for **ctx**.

SSL_add1_to_CA_list() appends the CA subject name extracted from **x** to the list of CAs sent to the peer for **s**, overriding the setting in the parent **SSL_CTX**.

NOTES

When a TLS/SSL server requests a client certificate (see **SSL_CTX_set_verify(3)**), it sends a list of CAs, for which it will accept certificates, to the client.

This list must explicitly be set using **SSL_CTX_set_client_CA_list()** or **SSL_CTX_set0_CA_list()** for **ctx** and **SSL_set_client_CA_list()** or **SSL_set0_CA_list()** for the specific **ssl**. The list specified overrides the previous setting. The CAs listed do not become trusted (**list** only contains the names, not the complete certificates); use **SSL_CTX_load_verify_locations**(3) to additionally load them for verification.

If the list of acceptable CAs is compiled in a file, the **SSL_load_client_CA_file**(3) function can be used to help to import the necessary data.

SSL_CTX_add_client_CA(), **SSL_CTX_add1_to_CA_list()**, **SSL_add_client_CA()** and **SSL_add1_to_CA_list()** can be used to add additional items the list of CAs. If no list was specified before using **SSL_CTX_set_client_CA_list()**, **SSL_CTX_set0_CA_list()**, **SSL_set_client_CA_list()** or **SSL_set0_CA_list()**, a new CA list for **ctx** or **ssl** (as appropriate) is opened.

RETURN VALUES

SSL_CTX_set_client_CA_list(), **SSL_set_client_CA_list()**, **SSL_CTX_set_client_CA_list()**, **SSL_set_client_CA_list()**, **SSL_CTX_set0_CA_list()** and **SSL_set0_CA_list()** do not return a value.

SSL_CTX_get_client_CA_list(), **SSL_get_client_CA_list()**, **SSL_CTX_get0_CA_list()** and **SSL_get0_CA_list()** return a stack of CA names or **NULL** is no CA names are set.

SSL_CTX_add_client_CA(),**SSL_add_client_CA()**, **SSL_CTX_add1_to_CA_list()** and **SSL_add1_to_CA_list()** return 1 for success and 0 for failure.

SSL_get0_peer_CA_list() returns a stack of CA names sent by the peer or **NULL** or an empty stack if no list was sent.

EXAMPLES

Scan all certificates in **CAfile** and list them as acceptable CAs:

```
SSL_CTX_set_client_CA_list(ctx, SSL_load_client_CA_file(CAfile));
```

SEE ALSO

ssl(7), **SSL_load_client_CA_file**(3), **SSL_CTX_load_verify_locations**(3)

COPYRIGHT