NAME

SSL_CTX_set1_groups, SSL_CTX_set1_groups_list, SSL_set1_groups, SSL_set1_groups_list, SSL_get1_groups, SSL_get_shared_group, SSL_get_negotiated_group, SSL_CTX_set1_curves, SSL_CTX_set1_curves_list, SSL_set1_curves, SSL_set1_curves_list, SSL_get1_curves, SSL_get_shared_curve - EC supported curve functions

SYNOPSIS

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

 int SSL_CTX_set1_groups(SSL_CTX *ctx, int *glist, int glistlen);
 int SSL_CTX_set1_groups_list(SSL_CTX *ctx, char *list);

 int SSL_set1_groups(SSL *ssl, int *glist, int glistlen);
 int SSL_set1_groups_list(SSL *ssl, char *list);

 int SSL_get1_groups(SSL *ssl, int *groups);
 int SSL_get_shared_group(SSL *s, int n);
 int SSL_get_negotiated_group(SSL *s);

 int SSL_CTX_set1_curves(SSL_CTX *ctx, int *clist, int clistlen);
 int SSL_CTX_set1_curves_list(SSL_CTX *ctx, char *list);

 int SSL_set1_curves(SSL *ssl, int *clist, int clistlen);
 int SSL_set1_curves_list(SSL *ssl, char *list);

 int SSL_get1_curves(SSL *ssl, int *curves);
 int SSL_get_shared_curve(SSL *s, int n);
```

DESCRIPTION

For all of the functions below that set the supported groups there must be at least one group in the list. A number of these functions identify groups via a unique integer NID value. However, support for some groups may be added by external providers. In this case there will be no NID assigned for the group. When setting such groups applications should use the "list" form of these functions (i.e. **SSL_CTX_set1_groups_list()** and SSL_set1_groups_list).

SSL_CTX_set1_groups() sets the supported groups for **ctx** to **glistlen** groups in the array **glist**. The array consist of all NIDs of groups in preference order. For a TLS client the groups are used directly in the supported groups extension. For a TLS server the groups are used to determine the set of shared groups. Currently supported groups for **TLSv1.3** are **NID_X9_62_prime256v1**, **NID_secp384r1**, **NID_secp521r1**, **NID_X25519**, **NID_X448**, **NID_ffdhe2048**, **NID_ffdhe3072**, **NID_ffdhe4096**,

NID_ffdhe6144 and **NID_ffdhe8192**.

SSL_CTX_set1_groups_list() sets the supported groups for **ctx** to string **list**. The string is a colon separated list of group NIDs or names, for example "P-521:P-384:P-256:X25519:ffdhe2048". Currently supported groups for **TLSv1.3** are **P-256**, **P-384**, **P-521**, **X25519**, **X448**, **ffdhe2048**, **ffdhe3072**, **ffdhe4096**, **ffdhe6144**, **ffdhe8192**. Support for other groups may be added by external providers.

SSL_set1_groups() and **SSL_set1_groups_list()** are similar except they set supported groups for the SSL structure **ssl**.

SSL_get1_groups() returns the set of supported groups sent by a client in the supported groups extension. It returns the total number of supported groups. The **groups** parameter can be **NULL** to simply return the number of groups for memory allocation purposes. The **groups** array is in the form of a set of group NIDs in preference order. It can return zero if the client did not send a supported groups extension. If a supported group NID is unknown then the value is set to the bitwise OR of TLSEXT_nid_unknown (0x1000000) and the id of the group.

SSL_get_shared_group() returns the NID of the shared group **n** for a server-side SSL **ssl**. If **n** is -1 then the total number of shared groups is returned, which may be zero. Other than for diagnostic purposes, most applications will only be interested in the first shared group so **n** is normally set to zero. If the value **n** is out of range, NID_undef is returned. If the NID for the shared group is unknown then the value is set to the bitwise OR of TLSEXT_nid_unknown (0x1000000) and the id of the group.

SSL_get_negotiated_group() returns the NID of the negotiated group used for the handshake key exchange process. For TLSv1.3 connections this typically reflects the state of the current connection, though in the case of PSK-only resumption, the returned value will be from a previous connection. For earlier TLS versions, when a session has been resumed, it always reflects the group used for key exchange during the initial handshake (otherwise it is from the current, non-resumption, connection). This can be called by either client or server. If the NID for the shared group is unknown then the value is set to the bitwise OR of TLSEXT_nid_unknown (0x1000000) and the id of the group.

All these functions are implemented as macros.

The curve functions are synonyms for the equivalently named group functions and are identical in every respect. They exist because, prior to TLS1.3, there was only the concept of supported curves. In TLS1.3 this was renamed to supported groups, and extended to include Diffie Hellman groups. The group functions should be used in preference.

NOTES

If an application wishes to make use of several of these functions for configuration purposes either on a command line or in a file it should consider using the SSL_CONF interface instead of manually parsing options.

RETURN VALUES

SSL_CTX_set1_groups(), **SSL_CTX_set1_groups_list()**, **SSL_set1_groups()** and **SSL_set1_groups_list()**, return 1 for success and 0 for failure.

SSL_get1_groups() returns the number of groups, which may be zero.

SSL_get_shared_group() returns the NID of shared group **n** or NID_undef if there is no shared group **n**; or the total number of shared groups if **n** is -1.

When called on a client **ssl**, **SSL_get_shared_group()** has no meaning and returns -1.

SSL_get_negotiated_group() returns the NID of the negotiated group used for key exchange, or NID_undef if there was no negotiated group.

SEE ALSO

ssl(7), **SSL_CTX_add_extra_chain_cert**(3)

HISTORY

The curve functions were added in OpenSSL 1.0.2. The equivalent group functions were added in OpenSSL 1.1.1. The **SSL_get_negotiated_group()** function was added in OpenSSL 3.0.0.

COPYRIGHT

Copyright 2013-2021 The OpenSSL Project Authors. All Rights Reserved.

Licensed under the Apache License 2.0 (the "License"). You may not use this file except in compliance with the License. You can obtain a copy in the file LICENSE in the source distribution or at <https://www.openssl.org/source/license.html>.