NAME

SSL_CTX_set_info_callback,   SSL_CTX_get_info_callback,   SSL_set_info_callback,
SSL_get_info_callback - handle information callback for SSL connections

SYNOPSIS

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

void SSL_CTX_set_info_callback(SSL_CTX *ctx, void (*callback)());
void (*SSL_CTX_get_info_callback(const SSL_CTX *ctx))();

void SSL_set_info_callback(SSL *ssl, void (*callback)());
void (*SSL_get_info_callback(const SSL *ssl))();
```

DESCRIPTION

SSL_CTX_set_info_callback() sets the **callback** function, that can be used to obtain state information for
SSL objects created from **ctx** during connection setup and use. The setting for **ctx** is overridden from the
setting for a specific SSL object, if specified. When **callback** is NULL, no callback function is used.

SSL_set_info_callback() sets the **callback** function, that can be used to obtain state information for **ssl**
during connection setup and use. When **callback** is NULL, the callback setting currently valid for **ctx** is
used.

SSL_CTX_get_info_callback() returns a pointer to the currently set information callback function for **ctx**.

SSL_get_info_callback() returns a pointer to the currently set information callback function for **ssl**.

NOTES

When setting up a connection and during use, it is possible to obtain state information from the SSL/TLS
engine. When set, an information callback function is called whenever the state changes, an alert appears,
or an error occurs.

The callback function is called as **callback(SSL *ssl, int where, int ret)**. The **where** argument specifies
information about where (in which context) the callback function was called. If **ret** is 0, an error condition
occurred. If an alert is handled, SSL_CB_ALERT is set and **ret** specifies the alert information.

where is a bitmask made up of the following bits:

SSL_CB_LOOP
        Callback has been called to indicate state change inside a loop.

SSL_CB_EXIT
        Callback has been called to indicate error exit of a handshake function. (May be soft error with retry
        option for non-blocking setups.)

SSL_CB_READ
        Callback has been called during read operation.

SSL_CB_WRITE
        Callback has been called during write operation.

SSL_CB_ALERT
        Callback has been called due to an alert being sent or received.

SSL_CB_READ_ALERT (SSL_CB_ALERT|SSL_CB_READ)
SSL_CB_WRITE_ALERT (SSL_CB_ALERT|SSL_CB_WRITE)
SSL_CB_ACCEPT_LOOP (SSL_ST_ACCEPT|SSL_CB_LOOP)
SSL_CB_ACCEPT_EXIT (SSL_ST_ACCEPT|SSL_CB_EXIT)
SSL_CB_CONNECT_LOOP (SSL_ST_CONNECT|SSL_CB_LOOP)
SSL_CB_CONNECT_EXIT (SSL_ST_CONNECT|SSL_CB_EXIT)
SSL_CB_HANDSHAKE_START
        Callback has been called because a new handshake is started.

SSL_CB_HANDSHAKE_DONE 0x20
Callback has been called because a handshake is finished.

The current state information can be obtained using the *SSL_state_string(3)* family of functions.

The **ret** information can be evaluated using the *SSL_alert_type_string(3)* family of functions.

RETURN VALUES

SSL_set_info_callback() does not provide diagnostic information.

SSL_get_info_callback() returns the current setting.

EXAMPLES

The following example callback function prints state strings, information about alerts being handled and error messages to the **bio_err** BIO.

```
void apps_ssl_info_callback(SSL *s, int where, int ret)
{
const char *str;
int w;

w = where & ~SSL_ST_MASK;

if (w & SSL_ST_CONNECT) str = "SSL_connect";
else if (w & SSL_ST_ACCEPT) str = "SSL_accept";
else str = "undefined";

if (where & SSL_CB_LOOP)
{
BIO_printf(bio_err, "%s:%s\n", str, SSL_state_string_long(s));
}
else if (where & SSL_CB_ALERT)
{
str = (where & SSL_CB_READ) ? "read" : "write";
BIO_printf(bio_err, "SSL3 alert %s:%s:%s\n",
str,
SSL_alert_type_string_long(ret),
SSL_alert_desc_string_long(ret));
}
else if (where & SSL_CB_EXIT)
{
if (ret == 0)
BIO_printf(bio_err, "%s:failed in %s\n",
str, SSL_state_string_long(s));
else if (ret < 0)
{
BIO_printf(bio_err, "%s:error in %s\n",
str, SSL_state_string_long(s));
}
}
}
```

SEE ALSO

ssl(3) , *SSL_state_string(3)* , *SSL_alert_type_string(3)*

COPYRIGHT

Copyright 2001-2016 The OpenSSL Project Authors. All Rights Reserved.

Licensed under the OpenSSL license (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>.