## NAME

SSL_psk_server_cb_func, SSL_psk_find_session_cb_func, SSL_CTX_use_psk_identity_hint, SSL_use_psk_identity_hint, SSL_CTX_set_psk_server_callback, SSL_set_psk_server_callback, SSL_CTX_set_psk_find_session_callback, SSL_set_psk_find_session_callback - set PSK identity hint to use

## SYNOPSIS

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

typedef int (*SSL_psk_find_session_cb_func)(SSL *ssl,
                                            const unsigned char *identity,
                                            size_t identity_len,
                                            SSL_SESSION **sess);


void SSL_CTX_set_psk_find_session_callback(SSL_CTX *ctx,
                                           SSL_psk_find_session_cb_func cb);
void SSL_set_psk_find_session_callback(SSL *s, SSL_psk_find_session_cb_func cb);

typedef unsigned int (*SSL_psk_server_cb_func)(SSL *ssl,
                                               const char *identity,
                                               unsigned char *psk,
                                               unsigned int max_psk_len);

int SSL_CTX_use_psk_identity_hint(SSL_CTX *ctx, const char *hint);
int SSL_use_psk_identity_hint(SSL *ssl, const char *hint);

void SSL_CTX_set_psk_server_callback(SSL_CTX *ctx, SSL_psk_server_cb_func cb);
void SSL_set_psk_server_callback(SSL *ssl, SSL_psk_server_cb_func cb);
```

## DESCRIPTION

A server application wishing to use TLSv1.3 PSKs should set a callback using either **SSL_CTX_set_psk_find_session_callback()** or **SSL_set_psk_find_session_callback()** as appropriate.

The callback function is given a pointer to the SSL connection in **ssl** and an identity in **identity** of length **identity_len**. The callback function should identify an SSL_SESSION object that provides the PSK details and store it in ***sess**. The SSL_SESSION object should, as a minimum, set the master key, the ciphersuite and the protocol version. See **SSL_CTX_set_psk_use_session_callback**(3) for details.

It is also possible for the callback to succeed but not supply a PSK. In this case no PSK will be used but

the handshake will continue. To do this the callback should return successfully and ensure that ***sess** is NULL.

Identity hints are not relevant for TLSv1.3. A server application wishing to use PSK ciphersuites for TLSv1.2 and below may call **SSL_CTX_use_psk_identity_hint()** to set the given **NUL**-terminated PSK identity hint **hint** for SSL context object **ctx**. **SSL_use_psk_identity_hint()** sets the given **NUL**-terminated PSK identity hint **hint** for the SSL connection object **ssl**. If **hint** is **NULL** the current hint from **ctx** or **ssl** is deleted.

In the case where PSK identity hint is **NULL**, the server does not send the ServerKeyExchange message to the client.

A server application wishing to use PSKs for TLSv1.2 and below must provide a callback function which is called when the server receives the ClientKeyExchange message from the client. The purpose of the callback function is to validate the received PSK identity and to fetch the pre-shared key used during the connection setup phase. The callback is set using the functions **SSL_CTX_set_psk_server_callback()** or **SSL_set_psk_server_callback()**. The callback function is given the connection in parameter **ssl**, **NUL**-terminated PSK identity sent by the client in parameter **identity**, and a buffer **psk** of length **max_psk_len** bytes where the pre-shared key is to be stored.

The callback for use in TLSv1.2 will also work in TLSv1.3 although it is recommended to use **SSL_CTX_set_psk_find_session_callback()** or **SSL_set_psk_find_session_callback()** for this purpose instead. If TLSv1.3 has been negotiated then OpenSSL will first check to see if a callback has been set via **SSL_CTX_set_psk_find_session_callback()** or **SSL_set_psk_find_session_callback()** and it will use that in preference. If no such callback is present then it will check to see if a callback has been set via **SSL_CTX_set_psk_server_callback()** or **SSL_set_psk_server_callback()** and use that. In this case the handshake digest will default to SHA-256 for any returned PSK. TLSv1.3 early data exchanges are possible in PSK connections only with the **SSL_psk_find_session_cb_func** callback, and are not possible with the **SSL_psk_server_cb_func** callback.

A connection established via a TLSv1.3 PSK will appear as if session resumption has occurred so that **SSL_session_reused**(3) will return true.

## RETURN VALUES

**SSL_CTX_use_psk_identity_hint()** and **SSL_use_psk_identity_hint()** return 1 on success, 0 otherwise.

Return values from the TLSv1.2 and below server callback are interpreted as follows:

0 PSK identity was not found. An "unknown_psk_identity" alert message will be sent and the connection setup fails.

>0 PSK identity was found and the server callback has provided the PSK successfully in parameter **psk**. Return value is the length of **psk** in bytes. It is an error to return a value greater than **max_psk_len**.

If the PSK identity was not found but the callback instructs the protocol to continue anyway, the callback must provide some random data to **psk** and return the length of the random data, so the connection will fail with decryption_error before it will be finished completely.

The **SSL_psk_find_session_cb_func** callback should return 1 on success or 0 on failure. In the event of failure the connection setup fails.

## NOTES

There are no known security issues with sharing the same PSK between TLSv1.2 (or below) and TLSv1.3. However, the RFC has this note of caution:

"While there is no known way in which the same PSK might produce related output in both versions, only limited analysis has been done. Implementations can ensure safety from cross-protocol related output by not reusing PSKs between TLS 1.3 and TLS 1.2."

## SEE ALSO

**ssl**(7), **SSL_CTX_set_psk_use_session_callback**(3), **SSL_set_psk_use_session_callback**(3)

## HISTORY

**SSL_CTX_set_psk_find_session_callback()** and **SSL_set_psk_find_session_callback()** were added in OpenSSL 1.1.1.

## COPYRIGHT