



Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'SSL_psk_client_cb_func.3oss1' command

`$ man SSL_psk_client_cb_func.3oss1`

`SSL_CTX_SET_PSK_CLIENT_CALLBACK(3ossOpenSSL_CTX_SET_PSK_CLIENT_CALLBACK(3oss1)`

NAME

SSL_psk_client_cb_func, SSL_psk_use_session_cb_func,
SSL_CTX_set_psk_client_callback, SSL_set_psk_client_callback,
SSL_CTX_set_psk_use_session_callback, SSL_set_psk_use_session_callback
- set PSK client callback

SYNOPSIS

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

```
typedef int (*SSL_psk_use_session_cb_func)(SSL *ssl, const EVP_MD *md,  
                                           const unsigned char **id,  
                                           size_t *idlen,  
                                           SSL_SESSION **sess);
```

```
void SSL_CTX_set_psk_use_session_callback(SSL_CTX *ctx,  
                                          SSL_psk_use_session_cb_func cb);
```

```
void SSL_set_psk_use_session_callback(SSL *s, SSL_psk_use_session_cb_func cb);
```

```
typedef unsigned int (*SSL_psk_client_cb_func)(SSL *ssl,  
                                               const char *hint,  
                                               char *identity,
```

```
unsigned int max_identity_len,  
unsigned char *psk,  
unsigned int max_psk_len);
```

```
void SSL_CTX_set_psk_client_callback(SSL_CTX *ctx, SSL_psk_client_cb_func cb);
```

```
void SSL_set_psk_client_callback(SSL *ssl, SSL_psk_client_cb_func cb);
```

DESCRIPTION

A client application wishing to use TLSv1.3 PSKs should use either `SSL_CTX_set_psk_use_session_callback()` or `SSL_set_psk_use_session_callback()` as appropriate. These functions cannot be used for TLSv1.2 and below PSKs.

The callback function is given a pointer to the SSL connection in `ssl`.

The first time the callback is called for a connection the `md` parameter is `NULL`. In some circumstances the callback will be called a second time. In that case the server will have specified a ciphersuite to use already and the PSK must be compatible with the digest for that ciphersuite. The digest will be given in `md`. The PSK returned by the callback is allowed to be different between the first and second time it is called.

On successful completion the callback must store a pointer to an identifier for the PSK in `*id`. The identifier length in bytes should be stored in `*idlen`. The memory pointed to by `*id` remains owned by the application and should be freed by it as required at any point after the handshake is complete.

Additionally the callback should store a pointer to an `SSL_SESSION` object in `*sess`. This is used as the basis for the PSK, and should, at a minimum, have the following fields set:

The master key

This can be set via a call to `SSL_SESSION_set1_master_key(3)`.

A ciphersuite

Only the handshake digest associated with the ciphersuite is relevant for the PSK (the server may go on to negotiate any ciphersuite which is compatible with the digest). The application can use any TLSv1.3 ciphersuite. If `md` is not NULL the handshake digest for the ciphersuite should be the same. The ciphersuite can be set via a call to `<SSL_SESSION_set_cipher(3)>`. The handshake digest of an `SSL_CIPHER` object can be checked using `<SSL_CIPHER_get_handshake_digest(3)>`.

The protocol version

This can be set via a call to `SSL_SESSION_set_protocol_version(3)` and should be `TLS1_3_VERSION`.

Additionally the maximum early data value should be set via a call to `SSL_SESSION_set_max_early_data(3)` if the PSK will be used for sending early data.

Alternatively an `SSL_SESSION` created from a previous non-PSK handshake may also be used as the basis for a PSK.

Ownership of the `SSL_SESSION` object is passed to the OpenSSL library and so it should not be freed by the application.

It is also possible for the callback to succeed but not supply a PSK.

In this case no PSK will be sent to the server but the handshake will continue. To do this the callback should return successfully and ensure that `*sess` is NULL. The contents of `*id` and `*idlen` will be ignored.

A client application wishing to use PSK ciphersuites for TLSv1.2 and

below must provide a different callback function. This function will be called when the client is sending the ClientKeyExchange message to the server.

The purpose of the callback function is to select the PSK identity and the pre-shared key to use during the connection setup phase.

The callback is set using functions `SSL_CTX_set_psk_client_callback()` or `SSL_set_psk_client_callback()`. The callback function is given the connection in parameter `ssl`, a NUL-terminated PSK identity hint sent by the server in parameter `hint`, a buffer identity of length `max_identity_len` bytes (including the NUL-terminator) where the resulting NUL-terminated identity is to be stored, and a buffer `psk` of length `max_psk_len` bytes where the resulting 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_use_session_callback()` or `SSL_set_psk_use_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_use_session_callback()` or `SSL_set_psk_use_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_client_callback()` or `SSL_set_psk_client_callback()` and use that. In this case the hint value will always be NULL and 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_use_session_cb_func` callback, and are not possible with the `SSL_psk_client_cb_func` callback.

NOTES

Note that parameter `hint` given to the callback may be NULL.

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.

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."

RETURN VALUES

Return values from the `SSL_psk_client_cb_func` callback are interpreted as follows:

On success (callback found a PSK identity and a pre-shared key to use) the length (> 0) of psk in bytes is returned.

Otherwise or on errors the callback should return 0. In this case the connection setup fails.

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

SEE ALSO

`ssl(7)`, `SSL_CTX_set_psk_find_session_callback(3)`,
`SSL_set_psk_find_session_callback(3)`

HISTORY

`SSL_CTX_set_psk_use_session_callback()` and
`SSL_set_psk_use_session_callback()` were added in OpenSSL 1.1.1.

COPYRIGHT

Copyright 2006-2021 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>.

3.0.7 2023-OSSL_CTX_SET_PSK_CLIENT_CALLBACK(3ossl)