



*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\_CTX\_get\_tlsext\_status\_arg.3ossl' command**

**`$ man SSL_CTX_get_tlsext_status_arg.3ossl`**

`SSL_CTX_SET_TLSEXT_STATUS_CB(3ossl) OpenSSLSSL_CTX_SET_TLSEXT_STATUS_CB(3ossl)`

### NAME

SSL\_CTX\_set\_tlsext\_status\_cb, SSL\_CTX\_get\_tlsext\_status\_cb,  
SSL\_CTX\_set\_tlsext\_status\_arg, SSL\_CTX\_get\_tlsext\_status\_arg,  
SSL\_CTX\_set\_tlsext\_status\_type, SSL\_CTX\_get\_tlsext\_status\_type,  
SSL\_set\_tlsext\_status\_type, SSL\_get\_tlsext\_status\_type,  
SSL\_get\_tlsext\_status\_ocsp\_resp, SSL\_set\_tlsext\_status\_ocsp\_resp - OCSP  
Certificate Status Request functions

### SYNOPSIS

```
#include <openssl/tls1.h>
```

```
long SSL_CTX_set_tlsext_status_cb(SSL_CTX *ctx, int (*callback)(SSL *, void *));
```

```
long SSL_CTX_get_tlsext_status_cb(SSL_CTX *ctx, int (**callback)(SSL *, void *));
```

```
long SSL_CTX_set_tlsext_status_arg(SSL_CTX *ctx, void *arg);
```

```
long SSL_CTX_get_tlsext_status_arg(SSL_CTX *ctx, void **arg);
```

```
long SSL_CTX_set_tlsext_status_type(SSL_CTX *ctx, int type);
```

```
long SSL_CTX_get_tlsext_status_type(SSL_CTX *ctx);
```

```
long SSL_set_tlsext_status_type(SSL *s, int type);
```

```
long SSL_get_tlsext_status_type(SSL *s);
```

```
long SSL_get_tlsext_status_ocsp_resp(ssl, unsigned char **resp);
```

```
long SSL_set_tlsext_status_ocsp_resp(ssl, unsigned char *resp, int len);
```

## DESCRIPTION

A client application may request that a server send back an OCSP status response (also known as OCSP stapling). To do so the client should call the `SSL_CTX_set_tlsext_status_type()` function prior to the creation of any SSL objects. Alternatively an application can call the `SSL_set_tlsext_status_type()` function on an individual SSL object prior to the start of the handshake. Currently the only supported type is `TLSEXT_STATUSTYPE_ocsp`. This value should be passed in the `type` argument. Calling `SSL_CTX_get_tlsext_status_type()` will return the type `TLSEXT_STATUSTYPE_ocsp` previously set via `SSL_CTX_set_tlsext_status_type()` or `-1` if not set.

The client should additionally provide a callback function to decide what to do with the returned OCSP response by calling `SSL_CTX_set_tlsext_status_cb()`. The callback function should determine whether the returned OCSP response is acceptable or not. The callback will be passed as an argument the value previously set via a call to `SSL_CTX_set_tlsext_status_arg()`. Note that the callback will not be called in the event of a handshake where session resumption occurs (because there are no Certificates exchanged in such a handshake). The callback previously set via `SSL_CTX_set_tlsext_status_cb()` can be retrieved by calling `SSL_CTX_get_tlsext_status_cb()`, and the argument by calling `SSL_CTX_get_tlsext_status_arg()`.

On the client side `SSL_get_tlsext_status_type()` can be used to determine whether the client has previously called `SSL_set_tlsext_status_type()`. It will return `TLSEXT_STATUSTYPE_ocsp` if it has been called or `-1` otherwise. On the server side

SSL\_get\_tlsext\_status\_type() can be used to determine whether the client requested OCSP stapling. If the client requested it then this function will return TLSEXT\_STATUSTYPE\_ocsp, or -1 otherwise.

The response returned by the server can be obtained via a call to SSL\_get\_tlsext\_status\_ocsp\_resp(). The value \*resp will be updated to point to the OCSP response data and the return value will be the length of that data. Typically a callback would obtain an OCSP\_RESPONSE object from this data via a call to the d2i\_OCSP\_RESPONSE() function. If the server has not provided any response data then \*resp will be NULL and the return value from SSL\_get\_tlsext\_status\_ocsp\_resp() will be -1.

A server application must also call the SSL\_CTX\_set\_tlsext\_status\_cb() function if it wants to be able to provide clients with OCSP Certificate Status responses. Typically the server callback would obtain the server certificate that is being sent back to the client via a call to SSL\_get\_certificate(); obtain the OCSP response to be sent back; and then set that response data by calling SSL\_set\_tlsext\_status\_ocsp\_resp(). A pointer to the response data should be provided in the resp argument, and the length of that data should be in the len argument.

## RETURN VALUES

The callback when used on the client side should return a negative value on error; 0 if the response is not acceptable (in which case the handshake will fail) or a positive value if it is acceptable.

The callback when used on the server side should return with either SSL\_TLSEXT\_ERR\_OK (meaning that the OCSP response that has been set should be returned), SSL\_TLSEXT\_ERR\_NOACK (meaning that an OCSP response should not be returned) or SSL\_TLSEXT\_ERR\_ALERT\_FATAL (meaning that a fatal error has occurred).

SSL\_CTX\_set\_tlsext\_status\_cb(), SSL\_CTX\_set\_tlsext\_status\_arg(),  
SSL\_CTX\_set\_tlsext\_status\_type(), SSL\_set\_tlsext\_status\_type() and  
SSL\_set\_tlsext\_status\_ocsp\_resp() return 0 on error or 1 on success.

SSL\_CTX\_get\_tlsext\_status\_type() returns the value previously set by  
SSL\_CTX\_set\_tlsext\_status\_type(), or -1 if not set.

SSL\_get\_tlsext\_status\_ocsp\_resp() returns the length of the OCSP  
response data or -1 if there is no OCSP response data.

SSL\_get\_tlsext\_status\_type() returns TLSEXT\_STATUSTYPE\_ocsp on the  
client side if SSL\_set\_tlsext\_status\_type() was previously called, or  
on the server side if the client requested OCSP stapling. Otherwise -1  
is returned.

## SEE ALSO

ssl(7)

## HISTORY

The SSL\_get\_tlsext\_status\_type(), SSL\_CTX\_get\_tlsext\_status\_type() and  
SSL\_CTX\_set\_tlsext\_status\_type() functions were added in OpenSSL 1.1.0.

## COPYRIGHT

Copyright 2015-2016 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>>.