

NAME

CURLINFO_TLS_SESSION, CURLINFO_TLS_SSL_PTR – get TLS session info

SYNOPSIS

```
#include <curl/curl.h>
```

```
CURLcode curl_easy_getinfo(CURL *handle, CURLINFO_TLS_SSL_PTR,
                           struct curl_tlssessioninfo **session);
```

```
/* if you need compatibility with libcurl < 7.48.0 use
   CURLINFO_TLS_SESSION instead: */
```

```
CURLcode curl_easy_getinfo(CURL *handle, CURLINFO_TLS_SESSION,
                           struct curl_tlssessioninfo **session);
```

DESCRIPTION

Pass a pointer to a 'struct curl_tlssessioninfo *'. The pointer will be initialized to refer to a 'struct curl_tlssessioninfo *' that will contain an enum indicating the SSL library used for the handshake and a pointer to the respective internal TLS session structure of this underlying SSL library.

This option may be useful for example to extract certificate information in a format convenient for further processing, such as manual validation. Refer to the **LIMITATIONS** section.

```
struct curl_tlssessioninfo {
    curl_sslbackend backend;
    void *internals;
};
```

The *backend* struct member is one of the defines in the CURLSSLBACKEND_* series: CURLSSLBACKEND_NONE (when built without TLS support), CURLSSLBACKEND_AXTLS, CURLSSLBACKEND_CYASSL, CURLSSLBACKEND_DARWINSSL, CURLSSLBACKEND_GNUTLS, CURLSSLBACKEND_GSKIT, CURLSSLBACKEND_MBEDTLS, CURLSSLBACKEND_NSS, CURLSSLBACKEND_OPENSSL, CURLSSLBACKEND_POLARSSL or CURLSSLBACKEND_SCHANNEL. (Note that the OpenSSL forks are all reported as just OpenSSL here.)

The *internals* struct member will point to a TLS library specific pointer for the active ("in use") SSL connection, with the following underlying types:

```
GnuTLS
    gnutls_session_t

gskit    gsk_handle

NSS      PRFileDesc *

OpenSSL
    CURLINFO_TLS_SESSION: SSL_CTX *
    CURLINFO_TLS_SSL_PTR: SSL *
```

Since 7.48.0 the *internals* member can point to these other SSL backends as well:

```
axTLS    SSL *

mbedtls
    mbedtls_ssl_context *

PolarSSL
    ssl_context *
```

Secure Channel (WinSSL)
 CtxtHandle *

Secure Transport (DarwinSSL)
 SSLContext *

WolfSSL (formerly CyaSSL)
 SSL *

If the *internals* pointer is NULL then either the SSL backend is not supported, an SSL session has not yet been established or the connection is no longer associated with the easy handle (eg curl_easy_perform has returned).

LIMITATIONS

This option has some limitations that could make it unsafe when it comes to the manual verification of certificates.

This option only retrieves the first in-use SSL session pointer for your easy handle, however your easy handle may have more than one in-use SSL session if using FTP over SSL. That is because the FTP protocol has a control channel and a data channel and one or both may be over SSL. **Currently there is no way to retrieve a second in-use SSL session associated with an easy handle.**

This option has not been thoroughly tested with plaintext protocols that can be upgraded/downgraded to/from SSL: FTP, SMTP, POP3, IMAP when used with *CURLOPT_USE_SSL(3)*. Though you will be able to retrieve the SSL pointer, it's possible that before you can do that **data (including auth) may have already been sent over a connection after it was upgraded.**

Renegotiation. If unsafe renegotiation or renegotiation in a way that the certificate is allowed to change is allowed by your SSL library this may occur and the certificate may change, and **data may continue to be sent or received after renegotiation but before you are able to get the (possibly) changed SSL pointer,** with the (possibly) changed certificate information.

If you are using OpenSSL or wolfSSL then *CURLOPT_SSL_CTX_FUNCTION(3)* can be used to set a certificate verification callback in the CTX. That is safer than using this option to poll for certificate changes and doesn't suffer from any of the problems above. There is currently no way in libcurl to set a verification callback for the other SSL backends.

How are you using this option? Are you affected by any of these limitations? Please let us know by making a comment at <https://github.com/curl/curl/issues/685>

PROTOCOLS

All TLS-based

EXAMPLE

TODO

AVAILABILITY

Added in 7.48.0.

This option supersedes *CURLINFO_TLS_SESSION(3)* which was added in 7.34.0. This option is exactly the same as that option except in the case of OpenSSL.

RETURN VALUE

Returns CURLE_OK if the option is supported, and CURLE_UNKNOWN_OPTION if not.

SEE ALSO

curl_easy_getinfo(3), *curl_easy_setopt(3)*, *CURLINFO_TLS_SESSION(3)*,