NAME

CURLOPT_SSL_CTX_DATA − pointer passed to SSL context callback

SYNOPSIS

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

CURLcode curl_easy_setopt(CURL *handle, CURLOPT_SSL_CTX_DATA, void *pointer);
```

DESCRIPTION

Data *pointer* to pass to the ssl context callback set by the option *CURLOPT_SSL_CTX_FUNCTION(3)*, this is the pointer you get as third parameter.

DEFAULT

NULL

PROTOCOLS

This functionality affects all TLS based protocols: HTTPS, FTPS, IMAPS, POP3S, SMTPS etc.

This option works only with the following TLS backends: OpenSSL, mbedTLS and wolfSSL

EXAMPLE

```
/* OpenSSL specific */

#include <openssl/ssl.h>
#include <curl/curl.h>
#include <stdio.h>

static CURLcode sslctx_function(CURL *curl, void *sslctx, void *pointer)
{
  X509_STORE *store;
  X509 *cert = NULL;
  BIO *bio;
  char *mypem = pointer;
  /* get a BIO */
  bio = BIO_new_mem_buf(mypem, -1);
  /* use it to read the PEM formatted certificate from memory into an
   * X509 structure that SSL can use
   */
  PEM_read_bio_X509(bio, &cert, 0, NULL);
  if(!cert)
    printf("PEM_read_bio_X509 failed...\n");

  /* get a pointer to the X509 certificate store (which may be empty) */
  store = SSL_CTX_get_cert_store((SSL_CTX *)sslctx);

  /* add our certificate to this store */
  if(X509_STORE_add_cert(store, cert) == 0)
    printf("error adding certificate\n");

  /* decrease reference counts */
  X509_free(cert);
  BIO_free(bio);

  /* all set to go */
  return CURLE_OK;
}
```

```
int main(void)
{
  CURL *curl;
  CURLcode result;
  /* CA cert in PEM format, replace the XXXs */
  char *mypem =
    "-----BEGIN CERTIFICATE-----\n"
    "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX\n"
    "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX\n"
    "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX\n"
    "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX\n"
    "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX\n"
    "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX\n"
    "-----END CERTIFICATE-----\n";

  curl_global_init(CURL_GLOBAL_ALL);
  curl = curl_easy_init();

  curl_easy_setopt(curl, CURLOPT_SSLCERTTYPE, "PEM");
  curl_easy_setopt(curl, CURLOPT_SSL_VERIFYPEER, 1L);
  curl_easy_setopt(curl, CURLOPT_URL, "https://www.example.com/");

  curl_easy_setopt(curl, CURLOPT_SSL_CTX_FUNCTION, *sslctx_function);
  curl_easy_setopt(curl, CURLOPT_SSL_CTX_DATA, mypem);
  result = curl_easy_perform(curl);
  if(result == CURLE_OK)
    printf("*** transfer succeeded ***\n");
  else
    printf("*** transfer failed ***\n");

  curl_easy_cleanup(curl);
  curl_global_cleanup();
  return (int)result;
}
```

HISTORY

Added in 7.11.0 for OpenSSL, in 7.42.0 for wolfSSL, in 7.54.0 for mbedTLS.

AVAILABILITY

Added in curl 7.10.6

RETURN VALUE

CURLE_OK if supported; or an error such as:

CURLE_NOT_BUILT_IN − Not supported by the SSL backend

CURLE_UNKNOWN_OPTION

SEE ALSO

CURLOPT_SSLVERSION(3), **CURLOPT_SSL_CTX_FUNCTION**(3)