NAME

CURLOPT_USE_SSL − request using SSL / TLS for the transfer

SYNOPSIS

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

CURLcode curl_easy_setopt(CURL *handle, CURLOPT_USE_SSL, long level);
```

DESCRIPTION

Pass a long using one of the values from below, to make libcurl use your desired *level* of SSL for the transfer.

These are all protocols that start out plain text and get "upgraded" to SSL using the STARTTLS command.

This is for enabling SSL/TLS when you use FTP, SMTP, POP3, IMAP etc.

CURLUSESSL_NONE
: Do not attempt to use SSL.

CURLUSESSL_TRY
: Try using SSL, proceed as normal otherwise. Note that server may close the connection if the negotiation fails. This level is *insecure* and should be avoided since it allows the connection to remain unprotected.

CURLUSESSL_CONTROL
: Require SSL for the control connection or fail with *CURLE_USE_SSL_FAILED*. This level is partially *insecure* and should be avoided since it lets the data connection remain unprotected.

  This level is meant for FTP, since that is the only protocol with separate connections for control and data. If used for IMAP, POP3 or SMTP it equals *CURLUSESSL_ALL*.

CURLUSESSL_ALL
: Require SSL for all communication or fail with *CURLE_USE_SSL_FAILED*.

DEFAULT

CURLUSESSL_NONE

PROTOCOLS

This functionality affects ftp, imap, pop3 and smtp

EXAMPLE

```
int main(void)
{
 CURL *curl = curl_easy_init();
 if(curl) {
  CURLcode result;
  curl_easy_setopt(curl, CURLOPT_URL, "ftp://example.com/dir/file.ext");

  /* require use of SSL for this, or fail */
  curl_easy_setopt(curl, CURLOPT_USE_SSL, CURLUSESSL_ALL);

  /* Perform the request */
  result = curl_easy_perform(curl);
  curl_easy_cleanup(curl);
 }
}
```

HISTORY

This option was known as CURLOPT_FTP_SSL up to 7.16.4. Supported by LDAP since 7.81.0. Fully supported by the OpenLDAP backend only.

CURLUSESSL_* enums became *long* types in 8.13.0, prior to this version a *long* cast was necessary when passed to *curl_easy_setopt(3)*.

AVAILABILITY

Added in curl 7.17.0

RETURN VALUE

curl_easy_setopt(3) returns a CURLcode indicating success or error.

CURLE_OK (0) means everything was OK, non−zero means an error occurred, see *libcurl−errors(3)*.

SEE ALSO

CURLOPT_PROXY_SSLVERSION(3), **CURLOPT_SSLVERSION**(3), **CURLOPT_SSL_OPTIONS**(3)