NAME

CURLOPT_SSL_OPTIONS − SSL behavior options

SYNOPSIS

#include <curl/curl.h>

CURLcode curl_easy_setopt(CURL *handle, CURLOPT_SSL_OPTIONS, long bitmask);

DESCRIPTION

Pass a long with a bitmask to tell libcurl about specific SSL behaviors. Available bits:

CURLSSLOPT_ALLOW_BEAST
: Tells libcurl to not attempt to use any workarounds for a security flaw in the SSL3 and TLS1.0 protocols. If this option is not used or this bit is set to 0, the SSL layer libcurl uses may use a work−around for this flaw although it might cause interoperability problems with some (older) SSL implementations.

  WARNING: avoiding this work−around lessens the security, and by setting this option to 1 you ask for exactly that. This option is only supported for Secure Transport and OpenSSL.

CURLSSLOPT_NO_REVOKE
: Tells libcurl to disable certificate revocation checks for those SSL backends where such behavior is present. This option is only supported for Schannel (the native Windows SSL library), with an exception in the case of Windows' Untrusted Publishers block list which it seems cannot be bypassed.

CURLSSLOPT_NO_PARTIALCHAIN
: Tells libcurl to not accept "partial" certificate chains, which it otherwise does by default. This option fails the certificate verification if the chain ends with an intermediate certificate and not with a root cert.

  Works with OpenSSL and its forks (LibreSSL, BoringSSL, etc). (Added in 7.68.0)

  Works with Schannel if the user specified certificates to verify the peer. (Added in 8.15.0)

CURLSSLOPT_REVOKE_BEST_EFFORT
: Tells libcurl to ignore certificate revocation checks in case of missing or offline distribution points for those SSL backends where such behavior is present. This option is only supported for Schannel (the native Windows SSL library). If combined with *CURLSSLOPT_NO_REVOKE*, the latter takes precedence. (Added in 7.70.0)

CURLSSLOPT_NATIVE_CA
: Tell libcurl to use the operating system's native CA store for certificate verification. This option is independent of other CA certificate locations set at run time or build time. Those locations are searched in addition to the native CA store.

  Works with wolfSSL on Windows, Linux (Debian, Ubuntu, Gentoo, Fedora, RHEL), macOS, Android and iOS (added in 8.3.0); with GnuTLS (added in 8.5.0) and with OpenSSL and its forks (LibreSSL, BoringSSL, etc) on Windows (Added in 7.71.0).

  This works with Rustls on Windows, macOS, Android and iOS. On Linux it is equivalent to using the Mozilla CA certificate bundle. When used with Rustls _only_ the native CA store is consulted, not other locations set at run time or build time. (Added in 8.13.0)

CURLSSLOPT_AUTO_CLIENT_CERT
: Tell libcurl to automatically locate and use a client certificate for authentication, when requested by the server. This option is only supported for Schannel (the native Windows SSL library). Prior to 7.77.0 this was the default behavior in libcurl with Schannel. Since the server can request any certificate that supports client authentication in the OS certificate store it could be a privacy

violation and unexpected. (Added in 7.77.0)

CURLSSLOPT_EARLYDATA
: Tell libcurl to try sending application data as TLS1.3 early data. This option is supported for GnuTLS, wolfSSL, quictls and OpenSSL (but not BoringSSL or AWS−LC). It works on TCP and QUIC connections using ngtcp2. This option works on a best effort basis, in cases when it was not possible to send early data the request is resent normally post−handshake. This option does not work when using QUIC. (Added in 8.11.0 for GnuTLS and 8.13.0 for wolfSSL, quictls and OpenSSL)

DEFAULT

0

PROTOCOLS

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

All TLS backends support this option.

EXAMPLE

```
int main(void)
{
 CURL *curl = curl_easy_init();
 if(curl) {
  CURLcode result;
  curl_easy_setopt(curl, CURLOPT_URL, "https://example.com/");
  /* weaken TLS only for use with silly servers */
  curl_easy_setopt(curl, CURLOPT_SSL_OPTIONS,
                   CURLSSLOPT_ALLOW_BEAST | CURLSSLOPT_NO_REVOKE);
  result = curl_easy_perform(curl);
  curl_easy_cleanup(curl);
 }
}
```

HISTORY

CURLSSLOPT_* macros became *long* types in 8.15.0, prior to this version a *long* cast was necessary when passed to *curl_easy_setopt(3)*.

AVAILABILITY

Added in curl 7.25.0

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_SSL_OPTIONS(3), **CURLOPT_SSLVERSION**(3), **CURLOPT_SSL_CIPHER_LIST**(3)