

NAME

curl_global_sslset – select SSL backend to use

SYNOPSIS

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

```
CURLSslset curl_global_sslset(curl_sslbackend id,
                               const char *name,
                               const curl_ssl_backend ***avail);
```

DESCRIPTION

This function configures at runtime which SSL backend to use with libcurl. This function can only be used to select an SSL backend once, and it must be called **before** *curl_global_init(3)*.

The backend can be identified by the *id* (e.g. **CURLSSLBACKEND_OPENSSL**). The backend can also be specified via the *name* parameter for a case insensitive match (passing **CURLSSLBACKEND_NONE** as *id*). If both *id* and *name* are specified, the *name* is ignored.

If neither *id* nor *name* are specified, the function fails with **CURLSSLSET_UNKNOWN_BACKEND** and set the *avail* pointer to the NULL-terminated list of available backends. The available backends are those that this particular build of libcurl supports.

Since libcurl 7.60.0, the *avail* pointer is always set to the list of alternatives if non-NULL.

Upon success, the function returns **CURLSSLSET_OK**.

If the specified SSL backend is not available, the function returns **CURLSSLSET_UNKNOWN_BACKEND** and sets the *avail* pointer to a NULL-terminated list of available SSL backends. In this case, you may call the function again to try to select a different backend.

The SSL backend can be set only once. If it has already been set, a subsequent attempt to change it results in a **CURLSSLSET_TOO_LATE** getting returned.

This function is thread-safe since libcurl 7.84.0 if *curl_version_info(3)* has the **CURL_VERSION_THREADSAFE** feature bit set (most platforms).

If this is not thread-safe, you must not call this function when any other thread in the program (i.e. a thread sharing the same memory) is running. This does not only mean no other thread that is using libcurl.

Names

SSL backend names (case-insensitive): GnuTLS, mbedTLS, OpenSSL, Rustls, Schannel, wolfSSL

The name "OpenSSL" is used for all versions of OpenSSL and its associated forks/flavors in this function. OpenSSL, BoringSSL, LibreSSL, quictls and AmiSSL are all supported by libcurl, but in the eyes of *curl_global_sslset(3)* they are all called "OpenSSL". They all mostly provide the same API. *curl_version_info(3)* can return more specific info about the exact OpenSSL flavor and version number in use.

struct

```
typedef struct {
    curl_sslbackend id;
    const char *name;
} curl_ssl_backend;
```

```
typedef enum {
    CURLSSLBACKEND_NONE = 0,
    CURLSSLBACKEND_OPENSSL = 1, /* or one of its forks */
    CURLSSLBACKEND_GNUTLS = 2,
```

```

CURLSSLBACKEND_NSS = 3,
CURLSSLBACKEND_GSKIT = 5, /* deprecated */
CURLSSLBACKEND_POLARSSL = 6, /* deprecated */
CURLSSLBACKEND_WOLFSSL = 7,
CURLSSLBACKEND_SCHANNEL = 8,
CURLSSLBACKEND_SECURETRANSPORT = 9, /* deprecated */
CURLSSLBACKEND_AXTLS = 10, /* deprecated */
CURLSSLBACKEND_MBEDTLS = 11,
CURLSSLBACKEND_MESALINK = 12, /* deprecated */
CURLSSLBACKEND_BEARSSL = 13, /* deprecated */
CURLSSLBACKEND_RUSTLS = 14
} curl_sslbackend;

```

PROTOCOLS

This functionality affects all supported protocols

EXAMPLE

```

int main(void)
{
    int i;
    /* choose a specific backend */
    curl_global_sslset(CURLSSLBACKEND_WOLFSSL, NULL, NULL);

    /* list the available ones */
    const curl_ssl_backend **list;
    curl_global_sslset(CURLSSLBACKEND_NONE, NULL, &list);

    for(i = 0; list[i]; i++)
        printf("SSL backend #%d: '%s' (ID: %d)\n",
            i, list[i]->name, list[i]->id);
}

```

AVAILABILITY

Added in curl 7.56.0

RETURN VALUE

If this function returns *CURLSSLSET_OK*, the backend was successfully selected.

If the chosen backend is unknown (or support for the chosen backend has not been compiled into libcurl), the function returns *CURLSSLSET_UNKNOWN_BACKEND*.

If the backend had been configured previously, or if *curl_global_init(3)* has already been called, the function returns *CURLSSLSET_TOO_LATE*.

If this libcurl was built completely without SSL support, with no backends at all, this function returns *CURLSSLSET_NO_BACKENDS*.

SEE ALSO

curl_global_init(3), **libcurl(3)**