NAME

CURLOPT_HTTPPROXYTUNNEL − tunnel through HTTP proxy

SYNOPSIS

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

CURLcode curl_easy_setopt(CURL *handle, CURLOPT_HTTPPROXYTUNNEL, long tunnel);
```

DESCRIPTION

Set the **tunnel** parameter to 1L to make libcurl tunnel all operations through the HTTP proxy (set with *CURLOPT_PROXY(3)*). There is a big difference between using a proxy and to tunnel through it.

Tunneling means that an HTTP CONNECT request is sent to the proxy, asking it to connect to a remote host on a specific port number and then the traffic is passed through the proxy. Proxies tend to white−list specific port numbers it allows CONNECT requests to and often only port 80 and 443 are allowed.

To suppress proxy CONNECT response headers from user callbacks use *CURLOPT_SUPPRESS_CONNECT_HEADERS(3)*.

HTTP proxies can generally only speak HTTP (for obvious reasons), which makes libcurl convert non−HTTP requests to HTTP when using an HTTP proxy without this tunnel option set. For example, asking for an FTP URL and specifying an HTTP proxy makes libcurl send an FTP URL in an HTTP GET request to the proxy. By instead tunneling through the proxy, you avoid that conversion (that rarely works through the proxy anyway).

DEFAULT

0

PROTOCOLS

This functionality affects all supported protocols

EXAMPLE

```
int main(void)
{
  CURL *curl = curl_easy_init();
  if(curl) {
    CURLcode result;
    curl_easy_setopt(curl, CURLOPT_URL, "ftp://example.com/file.txt");
    curl_easy_setopt(curl, CURLOPT_PROXY, "http://127.0.0.1:80");
    curl_easy_setopt(curl, CURLOPT_HTTPPROXYTUNNEL, 1L);
    result = curl_easy_perform(curl);
    curl_easy_cleanup(curl);
  }
}
```

AVAILABILITY

Added in curl 7.3

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(3), **CURLOPT_PROXYPORT**(3), **CURLOPT_PROXYTYPE**(3)