NAME

CURLOPT_OPENSOCKETFUNCTION − set callback for opening sockets

SYNOPSIS

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

typedef enum  {
  CURLSOCKTYPE_IPCXN,  /* socket created for a specific IP connection */
  CURLSOCKTYPE_ACCEPT, /* socket created by accept() call */
  CURLSOCKTYPE_LAST    /* never use */
} curlsocktype;

struct curl_sockaddr {
  int family;
  int socktype;
  int protocol;
  unsigned int addrlen;
  struct sockaddr addr;
};

curl_socket_t opensocket_callback(void *clientp,
                                  curlsocktype purpose,
                                  struct curl_sockaddr *address);

CURLcode curl_easy_setopt(CURL *handle, CURLOPT_OPENSOCKETFUNCTION, opensocket_callback);
```

DESCRIPTION

Pass a pointer to your callback function, which should match the prototype shown above.

This callback function gets called by libcurl instead of the *socket(2)* call. The callback's *purpose* argument identifies the exact purpose for this particular socket: *CURLSOCKTYPE_IPCXN* is for IP based connections and *CURLSOCKTYPE_ACCEPT* is for sockets created after accept() - such as when doing active FTP. Future versions of libcurl may support more purposes.

The *clientp* pointer contains whatever user-defined value set using the *CURLOPT_OPENSOCKETDATA(3)* function.

The callback gets the resolved peer address as the *address* argument and is allowed to modify the address or refuse to connect completely. The callback function should return the newly created socket or *CURL_SOCKET_BAD* in case no connection could be established or another error was detected. Any additional *setsockopt(2)* calls can of course be done on the socket at the user's discretion. A *CURL_SOCKET_BAD* return value from the callback function will signal an unrecoverable error to libcurl and it will return *CURLE_COULDNT_CONNECT* from the function that triggered this callback. This return code can be used for IP address blacklisting.

If you want to pass in a socket with an already established connection, pass the socket back with this callback and then use *CURLOPT_SOCKOPTFUNCTION(3)* to signal that it already is connected.

DEFAULT

The default behavior is the equivalent of this:

```
return socket(addr->family, addr->socktype, addr->protocol);
```

PROTOCOLS

All

EXAMPLE

AVAILABILITY

Added in 7.17.1.

RETURN VALUE

Returns CURLE_OK if the option is supported, and CURLE_UNKNOWN_OPTION if not.

SEE ALSO

CURLOPT_OPENSOCKETDATA(3), **CURLOPT_SOCKOPTFUNCTION**(3), **CURLOPT_CLOSESOCKETFUNCTION**(3),