

Win32 Minimum Client Socket Transaction

This is the minimal WinAPI using winsock2.h

References:

- <https://learn.microsoft.com/en-us/windows/win32/winsock/using-secure-socket-extensions>
- <https://learn.microsoft.com/en-us/windows/win32/api/winsock2/nf-winsock2-wssocketeta>
- https://learn.microsoft.com/en-us/windows/win32/api/winsock2/ns-winsock2-wsaprotocol_infoa
- <https://learn.microsoft.com/en-us/windows/win32/api/ws2tcpip/nf-ws2tcpip-wsasetsocketsecurity>
- <https://learn.microsoft.com/en-us/windows/win32/api/winsock/nf-winsock-wsastartup>
- https://learn.microsoft.com/en-us/windows/win32/api/mstcpip/ne-mstcpip-socket_security_protocol

Requires linking against ws2_32

```
#include <winsock2.h>
#include <mstcpip.h>

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, PSTR lpCmdLine, int nCmdShow){
    WORD wVersionRequired = MAKEWORD(2, 2);
    WSADATA wsaData;
    WSAStartup(wVersionRequired, &wsaData);

    // Create a socket
    SOCKET socket = WSASocketA(AF_INET, SOCK_STREAM, IPPROTO_TCP, NULL, 0, 0);
    if(socket == INVALID_SOCKET){
        printf("Socket was invalid with error %ld\n", WSAGetLastError());
    }

    // Establish the peer
```

```

struct sockaddr_in peer;
peer.sin_family = AF_INET;
peer.sin_addr.s_addr = inet_addr("127.0.0.1");
peer.sin_port = htons(50007);

// Connect to a peer.
int result = WSAConnect(socket, (SOCKADDR*) &peer, sizeof(peer), NULL, NULL, NULL, NULL);
cprintf("Result: %ld\n", result);


// Send some data
// We use the WSABUF here.
WSABUF buffer = {0};
char* text = "This is some data. Hello.";
buffer.buf = text;
buffer.len = strlen(text) + 1;


// Send
DWORD dwBytesSent = 0;
result = WSASend(socket, &buffer, 1, &dwBytesSent, 0, NULL, NULL);
if (result == SOCKET_ERROR) {
    result = WSAGetLastError();
    cprintf("WSASend returned error %ld\n", result);
} else {
    cprintf("Sent %u bytes across the channel.\n", dwBytesSent);
}


// Receive some data
// Set the recipient buffer.
#define maximumRecvSize 4000
char* recvBuffer = (char*) valloc(maximumRecvSize);
buffer.len = maximumRecvSize;
buffer.buf = recvBuffer;


DWORD dataFlags = 0;
//Request the data.
result = WSAREcv(socket, &buffer, 1, &dwBytesSent, &dataFlags, NULL, NULL);
if (result == SOCKET_ERROR) {
    result = WSAGetLastError();
    cprintf("WSAREcv returned error %ld\n", result);
}

```

```
} else {  
    printf("Received %u bytes across the channel.\n", dwBytesSent);  
    printf("Data: %s\n", buffer.buf);  
}  
  
// Close the socket  
result = closesocket(socket);  
printf("Result: %ld\n", result);  
  
WSACleanup();  
  
printf("Socket: %u\n", socket);  
return 0;  
}
```

Revision #3

Created 7 August 2024 17:06:56 by lepus

Updated 7 August 2024 17:27:35 by lepus