Subject: Problem using TcpSockets
Posted by koldo on Wed, 05 Feb 2014 20:02:09 GMT
View Forum Message <> Reply to Message

Hello!

I have been playing successfully with sockets based in SocketServer ans SocketClient demos.

However I have had problems with a simple program that sends plain text messages ended with a '\n with:

char *buf = "Message\n";
send(socket, buf, strlen(buf), 0);

Looking to SocketServer demo, it seems that s.Accept(server) blocks the program the second time. I mean:
- 1st s.Accept(server) passes just when demo program begins to send messages
- s.GetLine() works properly
- 2st s.Accept(server) blocks the program

However, if s.GetLine() is called in a loop, it takes all the messages properly.

Is there any kind of protocol embedded in TcpSocket used the way is used in SocketServer/Client demos, that does not match with plain send() calls?

Subject: Re: Problem using TcpSockets
Posted by mirek on Mon, 10 Feb 2014 19:16:27 GMT
View Forum Message <> Reply to Message

koldo wrote on Wed, 05 February 2014 15:02Hello!

I have been playing successfully with sockets based in SocketServer ans SocketClient demos.

However I have had problems with a simple program that sends plain text messages ended with a '\n with:

char *buf = "Message\n";
send(socket, buf, strlen(buf), 0);

Looking to SocketServer demo, it seems that s.Accept(server) blocks the program the second time. I mean:
- 1st s.Accept(server) passes just when demo program begins to send messages
- s.GetLine() works properly
- 2st s.Accept(server) blocks the program

However, if s.GetLine() is called in a loop, it takes all the messages properly.

Is there any kind of protocol embedded in TcpSocket used the way is used in SocketServer/Client demos, that does not match with plain send() calls?

Just to be sure: Do you open a new connection to the server for second 'send'? (the whole testcase would be useful).

Mirek

Subject: Re: Problem using TcpSockets
Posted by nlneilson on Tue, 11 Feb 2014 01:15:58 GMT
View Forum Message <> Reply to Message

A testcase would be useful to see the problem.

"- 2st s.Accept(server) blocks the program"
I am curious how you do that.

Whenever the last return is called:
 return "y";
}
#endif
the connection is closed and then a new connection can be made.

If you are in a GetLine() loop then the connection is not broken so a new
 if(!s.Connect("127.0.0.1", 11811)) { with the same address ("127.0.0.1" or whatever) will fail with an error saying that address is in use.

Servers are often set up to accept multiple connections so a different address could be opened. But a basic client with just one address can have only one connection for that address and cannot be opened more than once at the same time.

My thinking may be wrong but I think that is correct.

Subject: Re: Problem using TcpSockets
Posted by koldo on Tue, 11 Feb 2014 08:14:39 GMT
View Forum Message <> Reply to Message

Hello guys!

Lets put an example:

- In one side there is a client that just does socket "send()" sending strings ended with '\n'.

- In the other side, there is a server based on SocketServer demo.

This is similar to SocketServer, and just shows the first line :

```
 for(;;) {
  TcpSocket s;
  if(s.Accept(server)) {
   Cout() << "Request from: " << s.GetPeerAddr() << '\n';
   Cout() << "Received: " << s.GetLine() << "\n";
  }
 }
```

This shows all the lines (although it does not stop the loop);

```
 for(;;) {
  TcpSocket s;
  if(s.Accept(server)) {
   Cout() << "Request from: " << s.GetPeerAddr() << '\n';
   while(true)
    Cout() << "Received: " << s.GetLine() << "\n";
  }
 }
```

Subject: Re: Problem using TcpSockets
Posted by mirek on Tue, 11 Feb 2014 08:45:59 GMT
View Forum Message <> Reply to Message

If that is the case (first code not working, second does), then the problem is that you do not close 'send' socket.

I mean, the first example would only work if you close the connection, then establish a new one for next line.

Mirek

Subject: Re: Problem using TcpSockets
Posted by koldo on Wed, 12 Feb 2014 09:13:10 GMT
View Forum Message <> Reply to Message

I understand. This happened to me with an external program I wanted to connect to.

In addition, I have to do a socket client in plain C that I wanted to connect to an U++ server. Would I have to close the connection every time I send a data stream?

Subject: Re: Problem using TcpSockets
Posted by mirek on Wed, 12 Feb 2014 09:50:09 GMT
View Forum Message <> Reply to Message

Actually, I think you get the whole socket design wrong. I suggest some reading on 'accept'.

Anyway, if client sends a multitude of text in separate 'send's, then yours second source is right. You can replace '(true)' with something else like (!socket.IsEof()).

Subject: Re: Problem using TcpSockets
Posted by nlneilson on Fri, 14 Feb 2014 10:41:15 GMT
View Forum Message <> Reply to Message

If you have a considerable amount of data to send you can have many sentences that end with \n which is just another character.

The end is \0
