boost::asio cleanly disconnecting

Asked 10 years, 8 months ago Active 7 years, 8 months ago Viewed 45k times

32

13

Sometimes boost::asio seems to disconnect before I want it to, i.e. before the server properly handles the disconnect. I'm not sure how this is possible because the client seems to think its fully sent the message, yet when the server emits the error its not even read the message header... During testing this only happens maybe 1 in 5 times, the server receives the client shut down message, and disconnects the client cleanly.

The error: "An existing connection was forcibly closed by the remote host"

The client disconnecting:

```
void disconnect()
{
    boost::system::error_code error;
    //just creates a simple buffer with a shutdown header
    boost::uint8_t *packet = createPacket(PC_SHUTDOWN,0);
    //sends it
    if(!sendBlocking(socket,packet,&error))
    {
        //didnt get here in my tests, so its not that the write failed...
        logWrite(LOG_ERROR,"server",
            std::string("Error sending shutdown message.\n")
            + boost::system::system_error(error).what());
    }

    //actaully disconnect
    socket.close();
    ioService.stop();
}
bool sendBlocking(boost::asio::ip::tcp::socket &socket,
    boost::uint8_t *data, boost::system::error_code* error)
{
    //get the length section from the message
    boost::uint16_t len = *(boost::uint16_t*)(data - 3);
    //send it
    asio::write(socket, asio::buffer(data-3,len+3),
        asio::transfer_all(), *error);
    deletePacket(data);
    return !(*error);
}
```

The server:

```
void Client::clientShutdown()
{
    //not getting here in problem cases
    disconnect();
    boost::uint16_t len, const boost::system::error_code& error)
{
    if(error)
    {
        //error handled here
        delete[] data;
        std::stringstream ss;
        ss << "Error recieving packet.\n";
        ss << logInfo() << "\n";
        ss << "Error: " << boost::system::system_error(error).what();
        logWrite(LOG_ERROR,"Client",ss.str());

        disconnect();
    }
    else
    {
        //call handlers based on type, most will then call startRead when
        //done to get the next packet. Note however, that clientShutdown
        //does not
        ...
    }
}



void startRead(boost::asio::ip::tcp::socket &socket, PacketHandler handler)
{
    boost::uint8_t *header = new boost::uint8_t[3];
```

c++ sockets tcp network-programming boost-asio

share improve this question follow

edited Jun 18 '10 at 8:25
GrahamS
8,660 ●6 ●43 ●61

asked Jan 2 '10 at 23:14
Fire Lancer
26.9k ●25 ●107 ●163

1 Did you ever find an answer to this? – GrahamS Aug 25 '16 at 9:19

add a comment

5 Answers

Active Oldest Votes

32

I think you should probably have a call to `socket.shutdown(boost::asio::ip::tcp::socket::shutdown_both, ec)` in there before the call to `socket.close()`.

The boost::asio documentation for basic_stream_socket::close states:

> For portable behaviour with respect to graceful closure of a connected socket, call shutdown() before closing the socket.

This should ensure that any pending operations on the socket are properly cancelled and any buffers are flushed prior to the call to socket.close.

share improve this answer follow

edited Jun 20 at 9:12
Community ♦
1 ● 1

answered Jun 18 '10 at 8:23
GrahamS
8,660 ● 6 ● 43 ● 61

1 I've had exactly the same issue as Fire Lancer, and this solved it for me, thanks. This should probably be the accepted answer. – Silverlan Aug 15 '15 at 11:48

add a comment

13

I have tried to do this with both the close() method and the shutdown() method

```
socket.shutdown(boost::asio::ip::tcp::socket::shutdown_both, ec)
```

The shutdown method is the best of the two. However, I find that using the destructor of the ASIO socket is the clean way to do it as ASIO takes care of it all for you. So your goal is to just let the socket fall out of scope. Now, you can do this easily using a shared_ptr and resetting the shared_ptr to a fresh socket or null. this will call the destructor of the ASIO socket and life is good.

share improve this answer follow

answered Jan 7 '13 at 23:53
William Symionow
670 ● 10 ● 9

Using `shared_ptr` does the job very well! – nabroyan Jul 1 '16 at 13:59

2 This is misleading. The destructor acts only as if by socket.close(ec), see stackoverflow.com/a/39823756/1889040. So you must manually call `socket.shutdown` to close properly. I can confirm that without `socket.shutdown` buffers(prior to the call of close) may not be flushed. – scinart Feb 28 '18 at 5:03

add a comment

5

Maybe this is what is happening:

- Client send disconnect packet
- Client shuts socket down
- Server read handler gets called, but there is an error associated with the shutdown packet because the socket is already closed.

I see in your read handlers, if there is an error, you never check to see if your shutdown packet is there. Maybe it is. Basically what I'm saying is maybe your client sometimes is able to send both the close and the shutdown packet before the server has a chance to process them separately.

share improve this answer follow

answered Jan 3 '10 at 1:13
Chris H
5,781 ● 4 ● 29 ● 48

"//error "thrown" here, len always = 0 in problem cases..." so its always read 0 bytes of the header, ie it hasn't read any of the packet... And sticking a Sleep(500) or something on the client isn't a good solution, because that still may not always be enough on slower networks and is a noticeable delay. – Fire Lancer Jan 3 '10 at 7:33

message), I'll just end up with the opposite problem of the server disconnecting on the client.... – Fire Lancer Jan 3 '10 at 16:01

Would you be able to use the disconnect error alone or do you really need the disconnect packet for something? – Chris H Jan 5 '10 at 22:48

add a comment

4

Use async_write() and put socket.close() inside of write handler. This will make sure packet is processed by boost asio and not neglected in the middle of processing (because of close() calls).

share improve this answer follow

answered Aug 30 '10 at 21:38

flamemyst

977 ● 6 ● 9

add a comment

4

I have a very similar issue. I believe it's related to Windows recycling connections. Is the following familiar?

- you get this error immediately upon starting the program but not after a connection is established?
- The error never happens if you wait more than 4 minutes before restarting your application?

The tcp specs specify that by default it should wait four minutes for the final acknowledgment when a tcp connection is closed. You can see these connections in FIN_WAIT state using netstat. The Windows OS detects when you try to connect to the exact same system and takes these partially closed connections and recycles them. Your second invocation of the program gets the 'closed' connection left behind by the first run. It gets the next acknowledge and then really closes.

share improve this answer follow

answered Aug 30 '10 at 21:48

Jay

12.2k ● 3 ● 34 ● 64

add a comment

Your Answer

Sign up using Google

Sign up using Facebook

Sign up using Email and Password

Post as a guest

Name

Email

Required, but never shown

Post Your Answer

By clicking “Post Your Answer”, you agree to our terms of service, privacy policy and cookie policy

Not the answer you're looking for? Browse other questions tagged c++ sockets tcp network-programming boost-asio or ask your own question.

Linked

10	Boost.Asio socket destructor closes connection?
0	Boost.Asio: socket::close not cancelling aysnc_read_some

Related

6	Boost::asio async_wait handler signature
0	C++ boost asio tcp socket reading old data
3	Boost ASIO TCP separation of messages
0	boost::asio tcp server disconnect issue
5	How to keep a boost asio async read active continuously
0	Delegate use of boost asio TCP/IP socket

Hot Network Questions

Which is correct, and why? "My friend hasn't /

- Is there historical precedent for the US Senate to confirm a SCOTUS nominee from a POTUS of their own party in an election year?
- Is there any evidence to support the claim that English grammar is unusually straightforward?
- Employer planning on making a change that I'm prepared to quit over. How should I tell manager?
- Why do oscilloscopes have such high sampling rates when the bandwidth is low?
- DAGs: instrumental and adjusted variables
- Could there be a "divorce duel" to death?
- Find the result of some assignment statements
- Is there a "common voltage reference point" for all the circuits that connect together?
- Chaos with Newton’s method
- Python molar mass calculator
- Does hillbilly slang fall under a type of English language and if not, what is it referred as if anything?
- Can employer legally stop paying time & 1/2 to exempt employee after stating in the offer that they would do so?
- SF novel where the second Mars expedition discovers the first turned into vampires
- Why "Giraffe" as a name for the animal?
- Height on passport
- Build the following symmetric array
- How to customize the shapes when rasterizing photos
- Trying to cross into Tijuana by foot without a passport?
- How does "Somewhere you are" authentication add further security?
- Source for repentance being illogical (or counter intuitive)
- What is better: to have a modal open instantly and then load its contents, or to load its contents and then open it?
- How do I evaluate a manager I have a negative opinion of?
- how to ask cat (and maybe grep ?) not to take into account a new line when it's inside double quotes?

Question feed