Subject: Problem with websocket connect method
Posted by shutalker on Fri, 10 Nov 2017 06:05:10 GMT
View Forum Message <> Reply to Message

I've got a problem with websocket connect method. When a server drops connection, a client is still waiting for something. So if I do blocking connect, my app execution will be blocked.
Here what I've found in source WebSocket.cpp:

...

```
if(socket->IsBlocking()) {

...

    StartConnect();
    while(opcode != READING_FRAME_HEADER)
        Do0();
}

...
```

As I can understand, in blocking mode connect will never leave while cycle until it will read response header from server. And there is no check whether the socket was closed or an error was happened

I use upp sources from git: 97e1f20
Compiler: GCC 5.4.0

Subject: Re: Problem with websocket connect method
Posted by mirek on Wed, 15 Nov 2017 07:40:26 GMT
View Forum Message <> Reply to Message

shutalker wrote on Fri, 10 November 2017 07:05I've got a problem with websocket connect method. When a server drops connection, a client is still waiting for something. So if I do blocking connect, my app execution will be blocked.
Here what I've found in source WebSocket.cpp:

...

```
if(socket->IsBlocking()) {

...
```

```
    StartConnect();
    while(opcode != READING_FRAME_HEADER)
        Do0();
}

...
```

As I can understand, in blocking mode connect will never leave while cycle until it will read response header from server. And there is no check whether the socket was closed or an error was happened

I use upp sources from git: 97e1f20
Compiler: GCC 5.4.0

You are right. Hopefully fixed (in trunk):

```
 if(socket->IsBlocking()) {
  if(!addrinfo.Execute(host, port)) {
   Error("Not found");
   return false;
  }
  LLOG("DNS resolved");
  StartConnect();
  while(opcode != READING_FRAME_HEADER) {
   Do0();
   if(IsError())
    return false;
  }
 }
```

As you can see, I have also added return value for blocking mode...

Mirek

Subject: Re: Problem with websocket connect method
Posted by shutalker on Mon, 20 Nov 2017 14:44:29 GMT
View Forum Message <> Reply to Message

Yes, it's working. Thank you for the reply!

Subject: Re: Problem with websocket connect method

Posted by shutalker on Thu, 23 Nov 2017 20:58:17 GMT
View Forum Message <> Reply to Message

It seems I've got another problem with nonblocking Connect method. When I'm trying to connect in nonblocking mode from a simple client to a simple server my app generates error message after I call Do():

```
ws.NonBlocking().Connect("ws://127.0.0.1:8888");
ws.Do();
```

In nonblocking mode Connect() just sets opcode variable in DNS state and makes no operations until Do() is called:

```
if(socket->IsBlocking())
    ...
else
    opcode = DNS;

return true;
```

In Do0 there is if-statement listed below:

```
if(socket->IsEof() && !(close_sent || close_received))
    Error("Socket has been closed unexpectedly");
```

And in the IsEof method there is checkup for socket's descriptor state:

```
bool TcpSocket::IsEof() const
{
    return is_eof && ptr == end || IsAbort() || !IsOpen() || IsError();
}
```

As far as I can understand IsOpen returns true since the socket was not opened. So I can't process DNS state and make connection in nonblocking mode.

I used upp sources from github: 993904e
Compiler: GCC 5.4.1

Subject: Re: Problem with websocket connect method
Posted by Oblivion on Fri, 24 Nov 2017 13:28:34 GMT
View Forum Message <> Reply to Message

Hello shutalker, and welcome!

Quote:It seems I've got another problem with nonblocking Connect method. When I'm trying to connect in nonblocking mode from a simple client to a simple server my app generates error message after I call Do():

```
ws.NonBlocking().Connect("ws://127.0.0.1:8888");
ws.Do();
```

Is this how you execute the non-blocking call? Because you shouldn't. You should put Do() in a separate loop.
In non-blocking mode Connect() method clears and re-assigns the internal variables (socket too!) and schedules the operation. It is Do() method that progresses it (in a loop). (If you also put the Connect in the same loop as Do(), you'll get erratic behaviour.

A crude example:

```
ws.NonBlocking()
  .Connect("127.0.0.1:8888");

while(ws.Do())
  ; // Do something here...
```

If you can provide a simple example, or the actual code part, may be I will be able to help you.

Best regards,
Oblivion

Subject: Re: Problem with websocket connect method
Posted by shutalker on Fri, 24 Nov 2017 20:02:42 GMT
View Forum Message <> Reply to Message

Hello Oblivion!
Thank you for the reply!

Here's main part of simple nonblocking WebSocket client:

```
...
WebSocket ws;

ws.NonBlocking().Connect("ws://127.0.0.1:8888");

while( true )
{
```

```
    ws.Do();

    if( ws.IsClosed() || ws.IsError() )
    {
 LOG( ws.GetErrorDesc() );
 break;
    }

    SocketWaitEvent event;

    ws.AddTo( event );
    event.Wait( 1000 );

    if( event[0] & WAIT_READ )
    {
        String response = ws.Receive();

     if( !response.IsEmpty() )
  LOG( response );
    }
    else if( event[0] & WAIT_WRITE )
 ws.SendText( AsString( Random() ) );
}
...
```

It's a training code)

When I'm trying to run it I get error message: "Socket has been closed unexpectedly".
Is it the correct way I manage my websocket?
Could you explain how should I exactly create connection in this case in nonblocking mode?

Subject: Re: Problem with websocket connect method
Posted by Oblivion on Fri, 24 Nov 2017 20:33:33 GMT
View Forum Message <> Reply to Message

Hello shutalker,

Quote:
When I'm trying to run it I get error message: "Socket has been closed unexpectedly".
Is it the correct way I manage my websocket?
Could you explain how should I exactly create connection in this case in nonblocking mode?

Curious. Your code seems to be fine. I don't see anything wrong. But I noticed that you are trying to connect to the server at port 8888.
Just to be sure: Are you using the server from upp/reference/AsyncWebSocket.cpp?
Because the error you encounter can happen when you try to connect to wrong port.
Maybe you forgot to change the servers listening port to 8888?

In AsyncWebSocket.cpp example, line 38:

```
 if(!server.Listen(12321, 5, false, true)) {
  LOG("Cannot open server socket for listening!");
  Exit(1);
 }
```

As you can see server is listening on port 12321.

Best regards,
Oblivion

Subject: Re: Problem with websocket connect method
Posted by shutalker on Fri, 24 Nov 2017 20:45:26 GMT
View Forum Message <> Reply to Message

Yes. I'm using AsyncWebSocket example. All changes I've done was removing ClientEmulation from async server and replacing listening port from 12321 to 8888. Unfortunately, the problem wasn't solved

---

Subject: Re: Problem with websocket connect method
Posted by Oblivion on Fri, 24 Nov 2017 22:05:37 GMT
View Forum Message <> Reply to Message

Quote:
Yes. I'm using AsyncWebSocket example. All changes I've done was removing ClientEmulation from async server and replacing listening port from 12321 to 8888. Unfortunately, the problem wasn't solved

Ah, in the meantime I got what is wrong with your code.

Let me try to briefly explain the problem (as far as I can see):

Before you can send or receive anything to the server, the client needs to be connected to the server, and that is generally done after a "handshake"
You see, your "non-blocking" code is simply interrupting that handshake phase, hence the connection is forced to be closed.
You are trying to send and receive data in the middle of connection process.

But the real problem I see is there is no way (at least not that I can find one) to know if the

connection (handshake) is successful, and the client is ready. E.g. In similar scenarios, I personally use a Do() method which returns a boolean value so that I can detect the "end" of any operation (be it successful or erroneous). But WebSocket::Do() doesn't return anything, so we can't know if the Connect() operation is finished (except on failures).

This is probably by design and Mirek can probably give you a better explanation.

I'm sorry that I cannot be very helpful this time.

Best regards,
Oblivion

Subject: Re: Problem with websocket connect method
Posted by Oblivion on Sat, 25 Nov 2017 12:58:37 GMT
View Forum Message <> Reply to Message

Hello,

I found the problem with connection.
It turned out that

1) IpAddrInfo::Start() is not called (WebSocket.cpp, ln. 94-5.

```
 if(!ipaddrinfo.Execute(host, port)) {
  Error("Not found");
  return false;
 }
 LLOG("DNS resolved");
 StartConnect();
 while(opcode != READING_FRAME_HEADER) {
  Do0();
  if(IsError())
   return false;
 }
}
else {
 opcode = DNS;
 ipaddrinfo.Start(host, port); // <-- Was missing.
}
return true;
```

2) In Do0(), socket is checked for Eof too early (It should be checked after DNS phase, WebSocket, ln. 345-6).

```
  prev_opcode = opcode;
  if(opcode != DNS)
   if(socket->IsEof() && !(close_sent || close_received))
    Error("Socket has been closed unexpectedly");
```

3) I also changed addrinfo to ipaddrinfo to avoid any nameclashes and errors (there is already an "addrinfo" structure)
4) A suggestion: We should check for DNS lookup errors too:

```
void WebSocket::Dns()
{
 if(ipaddrinfo.InProgress())
  return;
        if(!ipaddrinfo.GetResult()) {                   // We should check for lookup errors...
              Error("DNS lookup failed");                //
              return;
        }
 LLOG("DNS resolved");
 StartConnect();
}
```

5) Aonther suggestion: I still think that WebSocket::Do() should return a boolean value to indicate progress/finish. I couldn't find a reliable way to check if a non-blocking operation such as Connect() is successful. Currently the only way to break the loop of a non-blocking Connect() call seems to be to check for errors.

From my experience I can say that below code (or other variants of this) would work well on such situations:

```
   ws.NonBlocking().Connect("127.0.0.1:12321");
   while(ws.Do())
       ;
   if(ws.IsError())
      return;

   // Success, carry on (send/recv)...
```

You can find the patched files below.

Best regards,
Oblivion

File Attachments

1) WebSocketPatched.zip, downloaded 432 times

Subject: Re: Problem with websocket connect method
Posted by shutalker on Sun, 26 Nov 2017 12:03:50 GMT
View Forum Message <> Reply to Message

Hello!
Thank you very much!
I hope your patch will appear in release sources soon.

Quote:
From my experience I can say that below code (or other variants of this) would work well on such situations:

```
ws.NonBlocking().Connect("127.0.0.1:12321");
while(ws.Do())
    ;
if(ws.IsError())
  return;

// Success, carry on (send/recv)...
```

As far as I can see, it's the same to:

```
ws.Connect("127.0.0.1:12321");

if(ws.IsError())
  return;

ws.NonBlocking();
// Success, carry on (send/recv)...
```

I also have a suggestion: it would be great to check whether connect is successful via special

method (e.g. IsConnected()) that returns true after the client has received websocket header from server in "handshake" stage. Roughly like this:

```
void WebSocket::ResponseHeader()
{
 if(ReadHttpHeader()) {
  LLOG(data);
  if(ToLower(data).Find("upgrade: websocket") < 0) {
   Error("Invalid server response HTTP header");
   return;
  }
  LLOG("HTTP response header received");
  opcode = READING_FRAME_HEADER;

                isConnected = true;        // <-- "handshake" successful

  data.Clear();
 }
}


bool WebSocket::IsConnected()
{
    return ( !socket->IsEof() && isConnected );
}
```

Subject: Re: Problem with websocket connect method
Posted by mirek on Sun, 26 Nov 2017 18:03:53 GMT
View Forum Message <> Reply to Message

Patch applied, with small variation:

```
void WebSocket::Do0()
{
 int prev_opcode;
 do {
  prev_opcode = opcode;
  if(findarg(opcode, DNS, SSL_HANDSHAKE) < 0) {
   Output();
   if(socket->IsEof() && !(close_sent || close_received))
    Error("Socket has been closed unexpectedly");
  }
  if(IsError())
   return;
  switch(opcode) {
```

- SSL_HANDSHAKE is the same situation...

Other than that, I need to apologize for "undertesting" Connect - I simply thought that Connect is not really important, as WebSocket is intended to be Browser->Server interface and we are the server...

Quote:
I also have a suggestion: it would be great to check whether connect is successful via special method (e.g. IsConnected()) that returns true after the client has received websocket header from server in "handshake" stage. Roughly like this:

Why do not you just use IsOpen?

Mirek

Subject: Re: Problem with websocket connect method
Posted by uppjj on Fri, 16 Feb 2018 12:43:48 GMT
View Forum Message <> Reply to Message

Hello

I found a little bug on Websocket class, Pong and Close message require bit FIN, =>SendRaw(PONG, data) should be SendRaw(FIN|PONG, data) otherwise remote socket close connection with 1002 code (protocole error).

You should also add "void Ping(const String& data) { SendRaw(FIN|PING, data); }" useful to check connection

JJ

Subject: Re: Problem with websocket connect method
Posted by mirek on Sat, 17 Feb 2018 11:07:24 GMT
View Forum Message <> Reply to Message

Thanks applied.
