

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) Another 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/rcv)...

```

You can find the patched files below.

Best regards,
Oblivion

File Attachments

1) [WebSocketPatched.zip](#), downloaded 333 times
