

Re: TCP server stop receiving new connections

Source:

<http://www.tech-archive.net/Archive/Development/microsoft.public.win32.programmer.networks/2007-07/msg00082>

- *From:* "Alexander Nickolov" <agnickolov@xxxxxxxx>
 - *Date:* Thu, 12 Jul 2007 17:02:24 -0700
-

What do you do in your OnAccept handler? In particular, do you try to accept multiple sockets? A test to try is to reset the event mask of your listening socket each time you exit its OnAccept handler.

Also, as a reality check – make sure your message loop never gets stuck... E.g. when you break into the debugger, almost always you should end up in the message loop.

--

=====
Alexander Nickolov
Microsoft MVP [VC], MCSD
email: agnickolov@xxxxxxxx
MVP VC FAQ: <http://vcfaq.mvps.org>
=====

"Julián Rodríguez Bajo" <julian.rodriguez@xxxxxxxx> wrote in message news:ecZt4s6wHHA.1776@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

History continues bellow...

Julián Rodríguez Bajo escribió:

Ali escribió:

On Jul 10, 4:34 pm, Julián Rodríguez Bajo
<julian.rodrig...@xxxxxxxx> wrote:

Ali escribió:

On Jul 9, 1:06 am, Julián
Rodríguez Bajo
<julian.rodrig...@xxxxxxxx>
wrote:

Re: TCP server stop receiving new connections

Hi folks.
I have a
strange
problem in
my class
library used
by all our
client
and
server
applications.
There are
servers
listening on
different
ports for
different
purposes.
Sometimes
(when a lot
of new
clients try
to connect
to the same
server),
one of the
servers (not
always the
same one)
stops
accepting
new
incomming
connections,
but keeps
current
connections.
Server port
appears as
LISTENING,
but
OnAccept
never
arrives to
server. New
client
connections
receive a
WSAECONNREFUSED
error. In this
situation, if

Re: TCP server stop receiving new connections

I
call Accept
directly in a
test
application,
one
incoming
connection
is
accepted,
but no
messages
can be sent
client ->
server nor
server ->
client, and
after 15
seconds
connection
is dropped
because of a
custom
keep alive
mechanism.
Running
server and a
lot of client
applications
in
same
machine
fails the
same way.
Both client
and server
processes
use the
same class
library. The
library
is a
statically
linked MFC
based static
library.
Both
classes,
server
and client,
derive from

Re: TCP server stop receiving new connections

CAsyncSocket.
There are
no threads
involved.
I have tried
changing
nConnectionBacklog
parameter
to Listen,
but
that
does not
seems to
make any
effect.
Any
suggestion
would be
apreciated.
Best
regards.

--snip--

Both
classes,
server and
client,
derive from
CAsyncSocket.
There are
no threads
involved.

If you are not using threads
fro non-blocked sockets
then how you
honour the requests from
multiple clients?
ali

When a CAsyncSocket derived socket is
created, by default a
WSAAsyncSelect is called with FD_READ |
FD_WRITE | FD_OOB | FD_ACCEPT |
FD_CONNECT | FD_CLOSE event
notification mask. Every socket
notification
is received in an WM_SOCKET_NOTIFY
message. MFC dispatches every socket
notification to the corresponding virtual

Re: TCP server stop receiving new connections

method of the CAsyncSocket
object:

- * FD_READ --> Calls OnReceive function.
- * FD_WRITE --> Calls OnSend function.
- * FD_OOB --> Calls OnOutOfBandData function.
- * FD_ACCEPT --> Calls OnAccept function.
- * FD_CONNECT --> Calls OnConnect function.
- * FD_CLOSE --> Calls OnClose function.

So, there is no need to use multiple threads
to serve multiple clients.

Hmm, sounds good to me. Your architecture looks good and quite natural
BUT it seems you need to tweak with your MFC message pump. Anyway, you
can write a simple hook for WM_SOCKET_NOTIFY message to double check
your network part interface.

ali

ali

Good idea Ali. I would hook WM_SOCKET_NOTIFY to see what is going on.
Maybe WM_SOCKET_NOTIFY messages are received but not delivered correctly
by MFC framework.

Thanks for your suggestion.

After some testing, problem seems that CSocketWnd::OnSocketNotify is never called. CSocketWnd::OnSocketNotify calls CSocket::ProcessAuxQueue, which calls CAsyncSocket::DoCallBack which calls the corresponding callback function of the CAsyncSocket object:

CSocketWnd::OnSocketNotify -->

Re: TCP server stop receiving new connections

```
CSocket::ProcessAuxQueue -->  
CAsyncSocket::DoCallBack -->  
CAsyncSocket::On[Send|OutOfBandData|Accept|Connect|Close]
```

In file sockcore.cpp @ line 1140:

```
BEGIN_MESSAGE_MAP(CSocketWnd, CWnd)  
//{{AFX_MSG_MAP(CSocketWnd)  
ON_MESSAGE(WM_SOCKET_NOTIFY, OnSocketNotify)  
ON_MESSAGE(WM_SOCKET_DEAD, OnSocketDead)  
//}}AFX_MSG_MAP  
END_MESSAGE_MAP()
```

So, OnSocketNotify must be called for every WM_SOCKET_NOTIFY received by the CSocketWnd, but as stated above, connecting around 50 clients concurrently to my server, OnSocketNotify is never called. Even if I accept "manually" an incoming connection, my server does not receive FD_READ from clients.