

Re: [6.x] problem with AIO, non-blocking sockets on freebsd and IE7 on windows.

Re: [6.x] problem with AIO, non-blocking sockets on freebsd and IE7 on windows.

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/net/2007-06/msg00204.html>

- *From:* Andre Oppermann <andre@xxxxxxxxxxxxx>
 - *Date:* Mon, 25 Jun 2007 21:07:52 +0200
-

Julian Elischer wrote:

Chuck Swiger wrote:

On Jun 25, 2007, at 10:46 AM, John-Mark Gurney wrote:

It's not the correct behaviour if the only packet coming back is an Ack of the FIN (and a FIN) because in the real world, making IE7 throw an error screen is not an acceptable option. This is the sort of thing that gets FreeBSD thrown out on favour of "anything else".

Believe me, our customers are "NOT HAPPY" about this.

Instead of getting an "authorization required" page along with the opportunity to log in, they get an error, and no opportunity to log in, which makes the system unusable. Yes, Blame Microsoft, but we are breaking the TCP spec, not them. We need to fix this some how.

As bde mention, the bug is in the application... Even SUSv2 says:

When all file descriptors associated with a pipe or FIFO special file are closed, any data remaining in the pipe or FIFO will be discarded.

A TCP socket isn't the same thing as a named pipe or FIFO. SUSv2 isn't the most relevant standard; RFC-793 is...

Re: [6.x] problem with AIO, non-blocking sockets on freebsd and IE7 on windows.

Our own close(2) says:
on the last
close of a socket(2) associated naming information and
queued data are
discarded

So, failure of the application to ensure that all data is sent is
the
application's fault... bde alluded to a simple work around of
clearing
the non-blocking flag which will return close to the
"expected" (but
apparently broken) behavior of keeping the tcp socket around
till all
remaining data has been sent...

I must note that the code you quoted has been in FreeBSD
since 2.0.

...and the relevant part is section 3.5 (circa pg 37) and the TCP state diagram
on pg 23. Using non-blocking I/O does not mean one can suddenly shortcut
the FINWAIT-1 and FINWAIT-2 states before going into TIME_WAIT,
nor the 2 * MSL timeout before the TCP control block is allowed to go away.

Otherwise, you might end up sending a RST to a dup'ed packet like a stray
ACK, which seems to be almost exactly the problem at hand.

Yes.

In fact it is not even a STRAY ACK. It is the REQUIRED ACK that the client
MUST send on reception of the FIN.

everyone has made good points so I think I'll re-iterate and comment.

1/ Some feel the app should not use NON-blocking on the close.
For an event-loop driven program using AIO and non-blocking sockets,
this is not an option. If the socket is blocking, and the far end has died, then the blocking
action would leave the whole event-loop blocked for the CLOSE_WAIT_1 period (i.e. 2
minutes
by default). This is the behaviour that Bruce was trying to avoid when he made the socket
obey the non-blocking flag in the first place. Even a 2 SECOND block whenever a client dies
is not an acceptable delay on a server serving 2000 requests per second.

2/ As has been pointed out, there is a difference between the action seen from the system call
interface point of view, and the
"on the wire" point of view. These are not the same thing. On the wire
we need to abide by the TCP RFC. We are not, leading to IE7 problems.
(and possibly others we are not aware of).

Re: [6.x] problem with AIO, non-blocking sockets on freebsd and IE7 on windows.

The answer is to decouple the behaviour of the protocol from the behaviour of the socket. My suggestion is to put the protocol control block for the session into a time event queue, just as is done for TIME_WAIT and other states, and have it abide by the time set in the SO_LINGER socket option. Even if the socket itself is long gone.

This is my task for the next day or so.. I will present patches for 6.x. Andre may decide to handle it differently in 7.

I agree with you here. This behavior is the source of the many log messages of syncache. It should be changed. The tcpcb and socket can already run decoupled, you only have to change the test in the close case in tcp_usrreq.c

—

Andre

freebsd-net@xxxxxxxxxxx mailing list

<http://lists.freebsd.org/mailman/listinfo/freebsd-net>

To unsubscribe, send any mail to "freebsd-net-unsubscribe@xxxxxxxxxxx"