

Re: Code Alignment Problem

Source:

<http://www.tech-archive.net/Archive/Development/microsoft.public.development.device.drivers/2005-02/0297.html>

From: cristalink (*cristalink_at_nospam.nospam*)

Date: 02/04/05

Date: Sat, 5 Feb 2005 09:16:05 +1300

The environment does not really matter. The code generated by the compiler is substantially different in debug and release modes. Most likely you have a bug in your code, such as an uninitialized variable etc. There's a chance you encountered a bug in the compiler or a timing problem.

Review your code. Run Prefast from the latest DDK on your sources. Try Driver Verifier. Usually, you need to understand the assembly code to track down such a bug. If you cannot read assembly code, you better hire an expert.

--

<http://www.firestreamer.com> - NTBackup to DVD and DV
"Ken Allen" <kendrhyd@sympatico.ca> wrote in message
news:O2JV6PvCFHA.1296@TK2MSFTNGP10.phx.gbl...
> We have an odd problem with one of our device drivers, and it seems to
> be associated with the placement and size of the code, but we cannot
> determine the exact cause or the correct resolution.
>
> First, I need to be clear that this is an older set of code, and it is
> being compiled with the VS 6 C++ compiler. The driver environment was
> hacked to permit the use of C++ code in the kernel, and so the
> environment is even more strange.
>
> There is one aspect of the driver that attempts to access the contents
> of a mountable file system during the boot process. Normally this works
> fine, as the attempt to open a file in that file system causes an
> auto-mount, and the file is accessed. We did notice that in debug mode
> the code was failing, and it turned out that the call to open the file
> was returning a "device not ready" error, which normally indicates that
> the file system could not be mounted.
>
> As we started to debug the cause of the problem, by adding extra debug
> information, the problem 'disappeared'! After several hours of
> experimenting with the code, we found that if we added or removed some
> code, we could cause the problem to occur or disappear! The actual code
> being added or removed does not seem to matter, as it is typically debug
> statements anyway, and in no way affects the algorithm.
>
> Does anyone have any suggestions on how we could go about isolating the
> cause for this problem and avoiding it in the future -- aside from
> rewriting the driver in the standard and latest DDK environment?
>
> -Ken