Why does a pointer change itself during function transition?

Asked 16 years, 2 months ago Modified 16 years, 2 months ago Viewed 2k times

5

In the following case I'm calling a `Func` with pointer passed to it, but in the called function, the parameter shows the pointer value as something totally bogus. Something like below.

```
bool flag = Func(pfspara);--> pfspara = 0x0091d910

bool Func(PFSPARA pfspara) --> pfspara = 0x00000005
{
    return false;
}
```

Why does `pfspara` change to some bogus pointer? I can't reproduce the problem in debug, only in production.

Thanks.

c++ pointers

Share
Improve this question
Follow

edited Oct 18, 2008 at 7:41

Daniel Spiewak
55.1k ● 14 ● 111 ● 120

asked Oct 18, 2008 at 6:57

Prache
543 ● 1 ● 6 ● 8

Could you post a small, complete example of a program that demonstrates this problem? It's hard to tell what might be going on by just code snippets. – Greg Hewgill Oct 18, 2008 at 6:58

I'm with you; defining the types is a bare minimum for this kind of question (up to and including function signatures). Also, is this C, C++, or something else? – Chris R Oct 18, 2008 at 7:13

I assumed c++ because of bool. If I am wrong, he can change the tag. – Bernard Oct 18, 2008 at 7:18

5 Answers

Sorted by: Highest score (default)

If you are trying to debug optimized code in for example Visual Studio, you cannot always rely on the debugger properly showing the values of variables - especially not

8

if the variable is unused so that the compiler probably optimizes it away.

Try running this instead:

```
bool Func(PFSPARA pfspara)
{
    printf("%x\n", pfspara);
    return false;
}
```

Share Improve this answer Follow

answered Oct 18, 2008 at 7:25

Rasmus Faber

49.6k ● 25 ● 148 ● 193

1

In general, this should never happen. Problems that can cause this type of symptoms include incompatibility in the compilation options between the calling and the caller modules, bad casting of member function pointers, or simply compiler bugs. You need to provide a lot more details about your problem: Show the real code, specify your compiler, specify what are the debug vs. production compilation flags, etc.

Share Improve this answer Follow

answered Oct 18, 2008 at 7:22

Oren Shemesh

1,568 ● 1 ● 8 ● 6

0

In addition to Rasmus' comments, I find it is generally worth checking whether the problem occurs in a debug build as well as the release build. If you see genuine problems occurring in a release build but not in the debug build, it is often down to a bug that is exposed by the optimization processs, such as an uninitialised variable. There is a real danger in doing most of your testing in a debug build, to avoid the problem you are seeing here, and then shipping a release build. IMO, if you don't have a good regression test suite (preferably automated) I would avoid shipping opimized code.

Share Improve this answer Follow

answered Oct 18, 2008 at 8:29

SmacL

22.9k ● 15 ● 99 ● 151

It sounds like a buffer overflow problem to me -- something is overwriting that variable. But as mentioned in other answers, there's no way to tell for sure without some actual code to work with.

0

Share Improve this answer Follow

answered Oct 18, 2008 at 20:35

Head Geek

39.8k • 22 • 79 • 89

It sounds to me like you're scribbling on the stack... somewhere in your code a buffer on the stack is overflowing, or you're taking the address of an object on the stack and writing to it after the function returns. This is causing your stack to be corrupted.

0

It may only happen in release mode because the stack allocations are different due to optimization and exclusion of 'guard' blocks used to help check for this kind of condition.

Share Improve this answer Follow

answered Oct 19, 2008 at 12:10

Zebra North

11.5k • 7 • 39 • 50