Subject: Crashes in IDL 7.1 & 8.0, interactive + VM mode, linux and Mac OS X
Posted by Stein Vidar Hagfors H on Wed, 19 Jan 2011 17:09:57 GMT
View Forum Message <> Reply to Message

I've found that a RESTORE of SAVE'd system variables often causes a crash [or failed assertion] inside glibc [on linux] and inside malloc [on OS X]. The crash doesn't occur immediately, it happens during subsequent actions at some point. I've been able to put together a modest tarball with code required to reproduce the error, but I've given up looking for the instructions on "this is how you report bugs to us so we can fix them, without going through your local provider or another few hoops".

So, I'm posting here [where I infrequently visit, nowadays, although I was 'big' back in the nineties ;-], in the hope that someone'll pick it up and run with it [maybe someone from VNI/RSI sees it].

The SAVE/RESTORE sequence can be right after one another, so it's not like the content of the sysvars should change...

Anyhow, here's a link to the tarball: http://folk.uio.no/steinhh/idl/crashing.tar.gz

Download, unzip/tar, make sure the directory ["crashing"] is writeable, cd into it, start IDL, do "@setup_crash".
After that, calling the procedure "crash" will save+restore system variables, and call a set of functions that'll provoke the crash. The setup_crash file also writes a "crash.sav" file that can be used to provoke the crash in VM mode (idl -vm=crash.sav). Hope this "works" for everyone else ;-)

I've realized that I'll need to find a workaround rather than wait for a fix, so I'm not going to take this any further...

Subject: Re: Crashes in IDL 7.1 & 8.0, interactive + VM mode, linux and Mac OS X
Posted by svhhaugan on Fri, 21 Jan 2011 15:51:20 GMT
View Forum Message <> Reply to Message

On Jan 19, 6:09 pm, svhhaugan <s.v.h.hau...@astro.uio.no> wrote:

> I've realized that I'll need to find a workaround rather than wait for
> a fix, so I'm not going to take this any further...

...and in doing so, I've narrowed down the cause to...<drum-roll>...

IDL> errstate = !error_state
IDL> !error_state = errstate
[See the crash below, starting with the "which,'blah" statement.

Thing is, it's impossible to SAVE,/SYSTEM_VARIABLES without getting this one bundled in there. But since I'm now doing them one-by-one using assignments to/from "normal" variables, it's easy to treat this one as yet-another variable-that-must-not-be-named. The other variable-that-must-not-be-named is !NULL, which is read-only, so one would not expect it to be assignable - the funny thing is that it doesn't give the same error message as assignments to e.g. !PI:

IDL> !pi=!pi
% Attempt to write to a readonly variable: !PI.
IDL> !null=!null
% Variable is undefined: <UNDEFINED>.

This kind'a makes sense, I suppose, but was still a big surprise to me 8-)

IDL> which,'blah
% Compiled module: WHICH.
% Compiled module: OS_FAMILY.
% Compiled module: HAVE_TAG.
% Compiled module: DELVARX.
% Compiled module: TRIM.
% Compiled module: GREP.
% Compiled module: APPEND_ARR.
% Compiled module: EXIST.
% Compiled module: STR_SEP.
% Compiled module: CONCAT_DIR.
*** glibc detected *** /astro/local/rsi/idl/bin/bin.linux.x86_64/idl:
free(): invalid pointer: 0x00002aee51788285 ***
======= Backtrace: =========
/lib64/libc.so.6[0x2aee5258a45f]
/lib64/libc.so.6(cfree+0x4b)[0x2aee5258a8bb]
/astro/local/rsi/idl/bin/bin.linux.x86_64/libidl.so.
8.0(IDL_MemFreeMSG_LONGJMP+0x39)[0x2aee50e9bb92]
======= Memory map: ========
00400000-00401000 r-xp 00000000 00:16
6406154 /mn/alruba/astro/local/rsi/idl80/
bin/bin.linux.x86_64/idl
00500000-00501000 rw-p 00000000 00:16
6406154 /mn/alruba/astro/local/rsi/idl80/
bin/bin.linux.x86_64/idl
0d175000-0d2c8000 rw-p 0d175000 00:00
0 [heap]
405b5000-405b6000 ---p 405b5000 00:00 0
405b6000-407b6000 rwxp 405b6000 00:00 0
40f30000-40f32000 rwxp 00000000 00:11
3148 /dev/zero

4102f000-41030000 ---p 4102f000 00:00 0
41030000-41230000 rwxp 41030000 00:00 0
415cb000-415cc000 ---p 415cb000 00:00 0
415cc000-417cc000 rwxp 415cc000 00:00 0
417cc000-417cd000 ---p 417cc000 00:00 0
417cd000-419cd000 rwxp 417cd000 00:00 0
35b9e00000-35b9e14000 r-xp 00000000 fd:02
1583063 /usr/lib64/libz.so.1.2.3
35b9e14000-35ba013000 ---p 00014000 fd:02
1583063 /usr/lib64/libz.so.1.2.3
35ba013000-35ba014000 rw-p 00013000 fd:02
1583063 /usr/lib64/libz.so.1.2.3
35ba200000-35ba305000 r-xp 00000000 fd:02
1583113 /usr/lib64/libX11.so.6.2.0
35ba305000-35ba505000 ---p 00105000 fd:02
1583113 /usr/lib64/libX11.so.6.2.0
35ba505000-35ba50c000 rw-p 00105000 fd:02
1583113 /usr/lib64/libX11.so.6.2.0
35ba600000-35ba605000 r-xp 00000000 fd:02
1583111 /usr/lib64/libXdmcp.so.6.0.0
35ba605000-35ba804000 ---p 00005000 fd:02
1583111 /usr/lib64/libXdmcp.so.6.0.0
35ba804000-35ba805000 rw-p 00004000 fd:02
1583111 /usr/lib64/libXdmcp.so.6.0.0
35baa00000-35baa02000 r-xp 00000000 fd:02
1583101 /usr/lib64/libXau.so.6.0.0
35baa02000-35bac01000 ---p 00002000 fd:02
1583101 /usr/lib64/libXau.so.6.0.0
35bac01000-35bac02000 rw-p 00001000 fd:02
1583101 /usr/lib64/libXau.so.6.0.0
35bae00000-35bae10000 r-xp 00000000 fd:02
1114113 /usr/lib64/libXext.so.6.4.0
35bae10000-35bb010000 ---p 00010000 fd:02
1114113 /usr/lib64/libXext.so.6.4.0
35bb010000-35bb011000 rw-p 00010000 fd:02
1114113 /usr/lib64/libXext.so.6.4.0
35be200000-35be217000 r-xp 00000000 fd:02
1114121 /usr/lib64/libICE.so.6.3.0
35be217000-35be416000 ---p 00017000 fd:02
1114121 /usr/lib64/libICE.so.6.3.0
35be416000-35be418000 rw-p 00016000 fd:02
1114121 /usr/lib64/libICE.so.6.3.0
35be418000-35be41b000 rw-p 35be418000 00:00 0
35be600000-35be609000 r-xp 00000000 fd:02
1114122 /usr/lib64/libSM.so.6.0.0
35be609000-35be809000 ---p 00009000 fd:02
1114122 /usr/lib64/libSM.so.6.0.0
35be809000-35be80a000 rw-p 00009000 fd:02

1114122 /usr/lib64/libSM.so.6.0.0
35bee00000-35bee17000 r-xp 00000000 fd:02
1114124 /usr/lib64/libXmu.so.6.2.0
35bee17000-35bf016000 ---p 00017000 fd:02
1114124 /usr/lib64/libXmu.so.6.2.0
35bf016000-35bf018000 rw-p 00016000 fd:02
1114124 /usr/lib64/libXmu.so.6.2.0
35bfa00000-35bfa10000 r-xp 00000000 fd:02
1114125 /usr/lib64/libXpm.so.4.11.0
35bfa10000-35bfc10000 ---p 00010000 fd:02
1114125 /usr/lib64/libXpm.so.4.11.0
35bfc10000-35bfc11000 rw-p 00010000 fd:02
1114125 /usr/lib64/libXpm.so.4.11.0
35c2800000-35c2899000 r-xp 00000000 fd:02
1114282 /usr/lib64/libGL.so.169.12
35c2899000-35c2998000 ---p 00099000 fd:02
1114282 /usr/lib64/libGL.Abort

Subject: Re: Crashes in IDL 7.1 & 8.0, interactive + VM mode, linux and Mac OS X
Posted by chris_torrence@NOSPAM on Thu, 27 Jan 2011 02:58:47 GMT
View Forum Message <> Reply to Message

Hi all,
I was able to get the following to crash IDL 8.0.1 on a 64-bit Mac
running OSX 10.6.6:

MacBook:bin chris$ idl
IDL Version 8.0.1 <Integration build Thu Jul 29 18:17:29 MDT 2010>,
Mac OS X (darwin x86_64 m64).
(c) 2010, ITT Visual Information Solutions
Installation number: 000015.
Licensed for use by: ITT ONLY

IDL> system_var_file = '/tmp/system.sav'
IDL> save,/system,file=system_var_file
IDL> restore,system_var_file
IDL> openr,1,'foo'
% OPENR: Error opening file. Unit: 1, File: foo
  No such file or directory
% Execution halted at: $MAIN$
IDL> openr,1,'foo'
idl(17500,0x7fff70ce2ca0) malloc: *** error for object 0x100603595:
pointer being freed was not allocated
*** set a breakpoint in malloc_error_break to debug
Abort trap

Notice that you have to call the "openr" twice before it crashes.

Can someone try the above code on their machine, and see if it also
fails?

I'll go ahead and log a bug, and we'll see what we can do. Thanks for
finding this!

-Chris
ITTVIS

Subject: Re: Crashes in IDL 7.1 & 8.0, interactive + VM mode, linux and Mac OS X
Posted by Michael Galloy on Thu, 27 Jan 2011 03:40:35 GMT
View Forum Message <> Reply to Message

On 1/26/11 7:58 PM, Chris Torrence wrote:
> Hi all,
> I was able to get the following to crash IDL 8.0.1 on a 64-bit Mac
> running OSX 10.6.6:
>
> MacBook:bin chris$ idl
> IDL Version 8.0.1<Integration build Thu Jul 29 18:17:29 MDT 2010>,
> Mac OS X (darwin x86_64 m64).
> (c) 2010, ITT Visual Information Solutions
> Installation number: 000015.
> Licensed for use by: ITT ONLY
>
> IDL> system_var_file = '/tmp/system.sav'
> IDL> save,/system,file=system_var_file
> IDL> restore,system_var_file
> IDL> openr,1,'foo'
> % OPENR: Error opening file. Unit: 1, File: foo
> No such file or directory
> % Execution halted at: $MAIN$
> IDL> openr,1,'foo'
> idl(17500,0x7fff70ce2ca0) malloc: *** error for object 0x100603595:
> pointer being freed was not allocated
> *** set a breakpoint in malloc_error_break to debug
> Abort trap
>
> Notice that you have to call the "openr" twice before it crashes.
>
> Can someone try the above code on their machine, and see if it also
> fails?
>
> I'll go ahead and log a bug, and we'll see what we can do. Thanks for
> finding this!

Yes, I get the same thing on IDL 8.0.1 on my Mac.

```
IDL> system_var_file = '/tmp/system.sav'
IDL> save,/system,file=system_var_file
IDL> restore,system_var_file
IDL> openr,1,'foo'
% OPENR: Error opening file. Unit: 1, File: foo
  No such file or directory
% Error occurred at: $MAIN$
% Execution halted at: $MAIN$
IDL> openr,1,'foo'
idl(7687,0x7fff710b8ca0) malloc: *** error for object 0x10060441d:
pointer being freed was not allocated
*** set a breakpoint in malloc_error_break to debug
Abort trap
```

Mike
--
www.michaelgalloy.com
Research Mathematician
Tech-X Corporation
