Ruby trunk - Bug #15766

Crash in 2.4, 2.5 involving at_exit

04/14/2019 03:17 PM - christian.boos@free.fr (Christian Boos)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:		**Backport:**	2.4: REQUIRED, 2.5: DONE, 2.6: DONTNEED

Description

Hello,

I've written a piece of code which reproducibly triggers a crash in some versions of Ruby (2.4 and 2.5, but not 2.2, 2.3 nor 2.6).

As this involves Minitest, I've started by filing a bug report on GitHub:

https://github.com/seattlerb/minitest/issues/789

However, as this triggers a crash, I've been advised to report it here as well. Also, I wasn't able to find a similar bug report.

I have not looked too deep, but it seems that the way Minitest deals with atexit handlers exposes a problem when there's a pending LocalJumpError.

You will find all the details in the Minitest bug 789, but let me summarize them again here:

```
require 'minitest/autorun'
require 'minitest/spec'

describe 'a simple crash reproducer' do
  before { return } # /!\
  it 'checks stuff' do
  end
end
```

Executing the above with the default minitest (or any newer version) won't work as a local return is not expected from within a before call (same thing for after).

But depending on the version, we may get a crash in addition to the error report:

- ruby 2.2.5p319 (2016-04-26 revision 54774) [x64-mingw32] => LocalJumpError: unexpected return (no crash)
- ruby 2.3.3p222 (2016-11-21 revision 56859) [x64-mingw32] => LocalJumpError: unexpected return (no crash)
- ruby 2.4.2p198 (2017-09-14 revision 59899) [x64-mingw32] => unexpected return (+ CRASH)
- ruby 2.4.5p335 (2018-10-18 revision 65137) [x64-mingw32] => unexpected return (+ CRASH)
- ruby 2.5.0p0 (2017-12-25 revision 61468) [x64-mingw32] => unexpected return (+ CRASH)
- ruby 2.5.3p105 (2018-10-18 revision 65156) [x64-mingw32] => unexpected return (+ CRASH)
- ruby 2.6.0p0 (2018-12-25 revision 66547) [x64-mingw32] => unexpected return (no crash)
- ruby 2.6.1p33 (2019-01-30 revision 66950) [x64-mingw32] => unexpected return (no crash)

I'm mainly developing on Windows, and use the pre-built versions from rubyinstaller.org,
but in order to be sure that this is *not* a Windows-specific issue, I also tried Ruby 2.5.0p0 and 2.5.1p57 on Linux and I get the same crash.

That crash always takes the same form:

```
D:/Workspace/install/windows64/ruby-2.4/lib/ruby/gems/2.4.0/gems/minitest-5.10.3/lib/minitest.rb:6
0: [BUG] Segmentation fault

-- Control frame information -----------------------------------------------
c:0003 p:---- s:0011 e:000010 CFUNC  :exit
c:0002 p:0027 s:0006 e:000005 BLOCK  D:/Workspace/install/windows64/ruby-2.4/lib/ruby/gems/2.4.0/g
ems/minitest-5.10.3/lib/minitest.rb:60 [FINISH]
```

```
c:0001 p:0000 s:0003 E:000520 (none) [FINISH]
```

As the problem doesn't happen with 2.6, I did hope that this problem would also be fixed in recent 2.4 and 2.5, but the problem is still present for 2.4.5p335 and 2.5.3p105 at least (latest versions from rubyinstaller.org).

Related issues:

Related to Ruby trunk - Bug #15282: Segfault (null pointer dereference) in va...	**Closed**

History

#1 - 04/14/2019 06:40 PM - christian.boos@free.fr (Christian Boos)

- Subject changed from Crash in 2.4, 2.5 involving atexit to Crash in 2.4, 2.5 involving at_exit

#2 - 04/14/2019 09:11 PM - MSP-Greg (Greg L)

Using a recent build (ruby 2.5.5p157 (2019-03-15 revision 67260) [x64-mingw32]), the code ran fine, or just like 2.6. I haven't checked on 2.4.

Using the current RubyInstaller release (ruby 2.5.3p105 (2018-10-18 revision 65156) [x64-mingw32]) I had the same behavior you reported, a SEGV.

Hence, it would seem that whatever issue is causing the SEGV, the fix has been backported...

#3 - 04/19/2019 01:38 PM - MSP-Greg (Greg L)

I can repo the issues with

ruby 2.4.6p354 (2019-04-01 revision 67394) [x64-mingw32]

2.5 & 2.6 do not SEGV.

#4 - 04/22/2019 11:37 AM - christian.boos@free.fr (Christian Boos)

On ruby_2_5, I bisected the fix to be ruby_2_5|r65582.

With r65581:

```
../Bug-15766.rb: unexpected return
C:/Dev/msys64-20161025/opt/ruby_2_5/lib/ruby/gems/2.5.0/gems/minitest-5.11.3/lib/minitest.rb:60: [BUG] Segment
ation fault
ruby 2.5.4p110 (2018-11-07 revision 65581) [x64-mingw32]
[...]
```

With r65582:

```
$ ruby -v
ruby 2.5.4p111 (2018-11-07 revision 65582) [x64-mingw32]

$ ruby ../Bug-15766.rb
../Bug-15766.rb: unexpected return
Run options: --seed 5588

# Running:
```

#5 - 04/23/2019 07:20 AM - nagachika (Tomoyuki Chikanaga)

- Related to Bug #15282: Segfault (null pointer dereference) in variable.c iv_index_tbl_make added

#6 - 04/23/2019 07:25 AM - nagachika (Tomoyuki Chikanaga)

- Backport changed from 2.4: UNKNOWN, 2.5: UNKNOWN, 2.6: UNKNOWN to 2.4: REQUIRED, 2.5: DONE, 2.6: DONTNEED

- Status changed from Open to Closed

Thank you for reporting and investigations.
I'll close this ticket because it was fixed on trunk and the stable maintainers watch Closed tickets for backport.
[Bug #15282] seems related.

#7 - 04/24/2019 07:23 PM - christian.boos@free.fr (Christian Boos)

It is indeed an exact duplicate of #15282, which you backported to 2.5.

However, you didn't backport it then to 2.4. I understand that now it's probably a bit too late for a backport.

Nevertheless, as I didn't see you closing this bug till now, I tried in the meantime to backport it myself (the fix for ruby_2_5 doesn't apply in a straightforward way to ruby_2_4).

In case anyone's interested, here's what I came up with:

```
diff --git a/eval.c b/eval.c
index 8be12ecd84..6a432ff4b7 100644
--- a/eval.c
+++ b/eval.c
@@ -502,14 +502,15 @@ setup_exception(rb_thread_t *th, int tag, volatile VALUE mesg, VALUE cause)
     else if (special_exception_p(th, mesg)) {
        mesg = ruby_vm_special_exception_copy(mesg);
     }
-    if (cause != Qundef) {
-       exc_setup_cause(mesg, cause);
-    }
-    else if (nocause) {
-       exc_setup_cause(mesg, Qnil);
+    if (cause == Qundef) {
+       if (nocause) {
+           cause = Qnil;
+       } else if (!rb_ivar_defined(mesg, id_cause)) {
+           cause = get_thread_errinfo(th);
+       }
     }
-    else if (!rb_ivar_defined(mesg, id_cause)) {
-       exc_setup_cause(mesg, get_thread_errinfo(th));
+    if (cause != Qundef && (NIL_P(cause) || !THROW_DATA_P(cause))) {
+       exc_setup_cause(mesg, cause);
     }

     file = rb_source_loc(&line);
```