Ruby master - Bug #16814
Segmentation fault in GC while running test/ruby/test_fiber.rb on s390x
04/24/2020 05:31 AM - ReiOdaira (Rei Odaira)

Status: Open
Priority: Normal
Assignee: 
Target version: ruby -v: ruby 2.8.0dev (2020-04-12T03:45:22Z master 5c27681813) [s390x-linux]
Backport: 2.5: UNKNOWN, 2.6: UNKNOWN, 2.7: UNKNOWN

Description
A segmentation fault almost always happens in test/ruby/test_fiber.rb with certain commits of latest Ruby on s390x.

$ make test-all TESTS=test/ruby/test_fiber.rb
Run options:
|--seed=90044
    "|--ruby=./miniruby -I./lib -I. -I.ext/common ./tool/runruby.rb --extout=.ext -- --disable-gems
    "|--excludes-dir=./test/excludes
    "|--name=!/memory_leak/

# Running tests:
[24/29] TestFiber#test_stack_size = 0.89 s
  1) Failure:
TestFiber#test_stack_size [/home/chkbuild/my-tmp/build/20200412T043305Z/ruby/test/ruby/test_fiber.rb:356]:
  pid 5713 killed by SIGABRT (signal 6) (core dumped)
  -e:1:in `print': stack level too deep (SystemStackError)
      from -e:1:in `rec'
      from -e:1:in `block (3 levels) in rec'
      from -e:1:in `times'
      from -e:1:in `block (2 levels) in rec'
      from -e:1:in `times'
      from -e:1:in `block in rec'
      from -e:1:in `times'
      from -e:1:in `block in rec'
      ... 172 levels...
      from -e:1:in `block in rec'
      -e: [BUG] Segmentation fault at 0x0000000000000000
ruby 2.8.0dev (2020-04-12T03:45:22Z master 5c27681813) [s390x-linux]

-- Control frame information ----------------------------------------------
c:0001 p:0000 s:0003 E:001e20 (none) [FINISH]

-- Other runtime information ----------------------------------------------
* Loaded script: -e

*_loaded features:
  0 enumerator.so
  1 thread.rb
  2 rational.so
  3 complex.so
  4 ruby2_keywords.rb
  5 /home/chkbuild/my-tmp/build/20200412T043305Z/ruby/.ext/s390x-linux/enc/encdb.so
  6 /home/chkbuild/my-tmp/build/20200412T043305Z/ruby/.ext/s390x-linux/enc/trans/transdb.so
  7 /home/chkbuild/my-tmp/build/20200412T043305Z/ruby/lib/rbconfig.rb
  8 /home/chkbuild/my-tmp/build/20200412T043305Z/ruby/lib/rubygems/compatibility.rb
  9 /home/chkbuild/my-tmp/build/20200412T043305Z/ruby/lib/rubygems/defaults.rb
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Address Range</th>
<th>Permissions</th>
<th>Access</th>
<th>Size</th>
<th>File Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>r-xp</td>
<td>00000000-00000000</td>
<td>00000000</td>
<td>5e:01</td>
<td>134626</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>r--p</td>
<td>00001000-00001000</td>
<td>00001000</td>
<td>5e:01</td>
<td>134626</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>rw-p</td>
<td>00002000-00002000</td>
<td>00002000</td>
<td>5e:01</td>
<td>134626</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>r-xp</td>
<td>00000000-00000000</td>
<td>00000000</td>
<td>5e:01</td>
<td>135803</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>r--p</td>
<td>00001000-00001000</td>
<td>00001000</td>
<td>5e:01</td>
<td>135803</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>rw-p</td>
<td>00002000-00002000</td>
<td>00002000</td>
<td>5e:01</td>
<td>135803</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>r-xp</td>
<td>00000000-00000000</td>
<td>00000000</td>
<td>5e:01</td>
<td>135760</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>r--p</td>
<td>00001000-00001000</td>
<td>00001000</td>
<td>5e:01</td>
<td>135760</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>rw-p</td>
<td>00002000-00002000</td>
<td>00002000</td>
<td>5e:01</td>
<td>135760</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>r-xp</td>
<td>00000000-00000000</td>
<td>00000000</td>
<td>5e:01</td>
<td>135797</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>r--p</td>
<td>00001000-00001000</td>
<td>00001000</td>
<td>5e:01</td>
<td>135797</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>rw-p</td>
<td>00002000-00002000</td>
<td>00002000</td>
<td>5e:01</td>
<td>135797</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>r-xp</td>
<td>00000000-00000000</td>
<td>00000000</td>
<td>5e:01</td>
<td>135793</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>r--p</td>
<td>00001000-00001000</td>
<td>00001000</td>
<td>5e:01</td>
<td>135793</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>rw-p</td>
<td>00002000-00002000</td>
<td>00002000</td>
<td>5e:01</td>
<td>135793</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>r-xp</td>
<td>00000000-00000000</td>
<td>00000000</td>
<td>5e:01</td>
<td>135793</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>r--p</td>
<td>00001000-00001000</td>
<td>00001000</td>
<td>5e:01</td>
<td>135793</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>rw-p</td>
<td>00002000-00002000</td>
<td>00002000</td>
<td>5e:01</td>
<td>135793</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>r-xp</td>
<td>00000000-00000000</td>
<td>00000000</td>
<td>5e:01</td>
<td>135793</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>r--p</td>
<td>00001000-00001000</td>
<td>00001000</td>
<td>5e:01</td>
<td>135793</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
<tr>
<td>rw-p</td>
<td>00002000-00002000</td>
<td>00002000</td>
<td>5e:01</td>
<td>135793</td>
<td>/usr/lib64/libfreebl3.so</td>
</tr>
</tbody>
</table>

05/10/2020
The segmentation fault happens in a Ruby script invoked from test_fiber.rb by EnvUtil.invoke_ruby(). The Ruby script is to deliberately cause a stack overflow as follows.

```ruby
$stdout.sync=true; def rec; print "."; 1.times{1.times{1.times{rec}}}; end; Fiber.new(rec).resume
```

On s390x, this script caused SystemStackError, which I think is expected. However, it was during the handling of the stack overflow when the segmentation fault happened.

The core dump shows the following stack trace.

```
(gdb) bt
#0 0x0000003ff62c3c350 in raise () from /lib64/libc.so.6
#1 0x0000003ff62c3d078 in abort () from /lib64/libc.so.6
#2 0x000002aa0c84bf9a in die () at error.c:646
#3 rb_bug_for_fatal_signal (default_sighandler=0x0, sig=sig@entry=11, 
ctx=ctx@entry=0x2aa2d18fc50, 
fmt=fmt@entry=0x2aa0c89352c "Segmentation fault at %p") at error.c:678
#4 0x000002aa0c6fbd60 in sigsegv (sig=optimized out, info=0x2aa2d18fc50, 
ctx=0x2aa2d18fc50) at signal.c:955
#5 <signal handler called>
#6 0x000002aa0c5d4f36 in gc_mark_children ( 
objspace=objspace@entry=0x2aa2d09c6b0, obj=obj@entry=2929923415200) 
at gc.c:5478
#7 0x000002aa0c5d3ac8 in rgen_gc_remembert_set_mark (heap=0x2aa2d09c6d8, 
objspace=0x2aa2d09c6b0) at gc.c:6747
#8 gc_marks_start (full_mark=optimized out), objspace=0x2aa2d09c6b0) 
at gc.c:6314
#9 gc_marks (full_mark=optimized out), objspace=0x2aa2d09c6b0) at gc.c:6583
#10 gc_start (objspace=objspace@entry=0x2aa2d09c6b0, reason=optimized out), 
(reason=entry=256) at gc.c:7370
#11 0x000002aa0c5a9370 in heap_prepare (heap=0x2aa2d09c6d8, 
objspace=optimized out) at gc.c:1977
#12 heap_get_freeobj_from_next_freeobj ( 
objspace=objspace@entry=0x2aa2d09c6b0, heap=heap@entry=0x2aa2d09c6d8) 
at gc.c:1989
```

---Type <return> to continue, or q <return> to quit---

```
#13 0x000002aa0c5d4f36 in heap_get_freeobj (heap=0x2aa2d09c6d8, 
objspace=0x2aa2d09c6b0) at gc.c:2028
#14 newobj_slowpath (wb_protected=1, objspace=0x2aa2d09c6b0, v3=v3@entry=0, 
v2=0, v1=0, flags=5, klass=292992363840) at gc.c:2170
#15 newobj_slowpath_wb_protected (klass=292992363840, flags=5, v1=v1@entry=0, 
v2=v2@entry=0, v3=v3@entry=0, objspace=0x2aa2d09c6b0) at gc.c:2182
#16 0x000002aa0c5d817c in newobj_of (wb_protected=1, v3=0, v2=0, v1=0, 
flags=5, klass=optimized out) at gc.c:2218
#17 rb_wb_protected_newobj_of (klass=optimized out), flags=flags@entry=5) 
at gc.c:2234
#18 0x000002aa0c5d4f36 in str_alloc (klass=optimized out) at string.c:745
#19 str_new0 (klass=optimized out), ptp=0x2aa0c85ad0e "\n", len=1, 
termlen=optimized out) at string.c:767
#20 0x000002aa0c5b1f28 in ruby_str_new_cstr (str=0x2aa0c85ad0e "\n"
) at /include/ruby/3/interm/string.h:159
#21 print_backtrace (eclass=2929923530680, errat=errat@entry=292992354920, 
str=str@entry=8, reverse=reverse@entry=0) at eval_error.c:250
#22 0x000002aa0c5b3f68 in print_backtrace (reverse=0, str=8, 
errat=optimized out), eclass=optimized out) at eval_error.c:233
#23 rb_error_write (reverse=0, highlight=0, str=8, errat=optimized out), 
emesg=2929923530600, errinfo=optimized out) at eval_error.c:340
```
At gc.c:5478, the segmentation fault happened because any->as.typeddata.type was 0. as.typeddata.type should not be 0 for RTypedData.

5473 case T_DATA:
5474 {
5475     void *const ptr = DATA_PTR(obj);
5476     if (ptr) {
5477         RUBY_DATA_FUNC mark_func = RTYPEDDATA_P(obj) ?
5478             any->as.typeddata.type->function.dmark :
5479             any->as.data.dmark;
5480         if (mark_func) (*mark_func)(ptr);
5481     }
5482 }
5483 break;

This is a timing bug, but it almost always happens with 5c27681813. It is not clear to which commit this issue is related. In Ruby CI, it started happening in early February 2020 and stopped showing up after increasing the stack size by ulimit -s. It started happening again in early April 2020 and disappeared on April 15.

Anybody has any ideas how I should debug this?

History
#1 - 04/24/2020 06:07 AM - nobu (Nobuyoshi Nakada)
Where does any->as.data.free point?
Is any->as.basic.klass a valid class object?
If you compile gc.c as make DEFS=-DGC_DEBUG gc.o, any->file and any->line have the location in ruby level, and could help you.

#2 - 04/24/2020 06:38 AM - mame (Yusuke Endoh)

disappeared on April 15.

You may know, but the test has been skipped on s390x since 9948addeda67f4b7a6e357f1ebe9025f998b11d2.

#3 - 04/24/2020 11:05 PM - ReiOdaira (Rei Odaira)
Did you mean any->as.data.dfree? It points to no valid location.

(gdb) print any->as.data
$4 = (basic = {flags = 12, klass = 2930849422520}, dmark = 0x0, dfree = 0x1,
     data = 0x2aa6449f9e0)
(gdb) print any->as.typeddata
$5 = (basic = {flags = 12, klass = 2930849422520}, type = 0x0, typed_flag = 1,
     data = 0x2aa6449f9e0)

any->as.basic.klass seems to be a valid class. Is there any way to figure out what class it is using the core dump file?

(gdb) print *((struct RBasic *)any->as.basic.klass
$7 = {flags = 98, klass = 2930849422480)
(gdb) print *((struct RBasic *)any->as.basic.klass)->flags & 0x1f
$9 = 2
I've tried make DEFS=-DGC_DEBUG gc.o. It made the test fail quite less often than before, and when it failed, it did at a different location in GC (gc.c:5240), but it will help a lot. Thanks.

Thanks, Endoh-san, I didn't know the test was skipped.
FYI: I re-enabled the test in question with 93ed465dcd866013cd93c3662937497900c8086