Ruby - Bug #4828

crash in test_thread_instance_variable

06/05/2011 01:52 PM - kosaki (Motohiro KOSAKI)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	-	**Backport:**	

Description

"make test-all" on MacOS X makes following failure. It is very frequently happen.

1. Failure:
   test_thread_instance_variable(TestThread) [/Users/kosaki/ruby/test/ruby/test_thread.rb:583]:
   [ruby-core:35192].
   <[]> expected but was
   <["ruby-193(10898,0x7fff70f38ca0) malloc: *** error for object 0x1006f6cb0: pointer being freed was not allocated",
   "*** set a breakpoint in malloc_error_break to debug"]>.

Process: ruby-193 [10898]
Path: /Users/kosaki/ruby/ruby-193
Identifier: ruby-193
Version: ??? (???)
Code Type: X86-64 (Native)
Parent Process: ruby-193 [10884]

Date/Time: 2011-06-05 13:49:04.919 +0900
OS Version: Mac OS X 10.6.7 (10J4138)
Report Version: 6

Exception Type: EXC_CRASH (SIGABRT)
Exception Codes: 0x0000000000000000, 0x0000000000000000
Crashed Thread: 0 Dispatch queue: com.apple.main-thread

Application Specific Information:
abort() called

Thread 0 Crashed: Dispatch queue: com.apple.main-thread
0 libSystem.B.dylib 0x00007fff885225d6 __kill + 10
1 libSystem.B.dylib 0x00007fff885c2cde abort + 83
2 libSystem.B.dylib 0x00007fff884da6b5 free + 128
3 ruby-193 0x000000010004f693 vm_xfree + 25
4 ruby-193 0x000000010004f8c6 ruby_xfree + 45
5 ruby-193 0x00000001001064b3 st_free_table + 34
6 ruby-193 0x0000000100180671 thread_free + 216
7 ruby-193 0x0000000100180015 ruby_vm_destruct + 90
8 ruby-193 0x000000010003d2d0 ruby_cleanup + 853
9 ruby-193 0x000000010003d4a5 ruby_run_node + 63
10 ruby-193 0x0000000100000daf main + 79 (main.c:40)
11 ruby-193 0x0000000100000d58 start + 52

Thread 0 crashed with X86 Thread State (64-bit):
rax: 0x0000000000000000 rbx: 0x0000000000000001 rcx: 0x00007fff5fbff4b8 rdx: 0x0000000000000000
rdi: 0x0000000000002a92 rsi: 0x0000000000000006 rbp: 0x00007fff5fbff4d0 rsp: 0x00007fff5fbff4b8
r8: 0x0000000000000e03 r9: 0x0000000000000000 r10: 0x00007fff8851e616 r11: 0xffffff80002e4730
r12: 0x00000001002eb000 r13: 0x000000010031b000 r14: 0x0000000000000000 r15: 0x00000001006f6cb0
rip: 0x00007fff885225d6 rfl: 0x0000000000000202 cr2: 0x000000010035e000

Binary Images:
0x100000000 - 0x100225ff7 +ruby-193 ??? (???) <0104E33C-47DB-D944-5F62-DB75E2831561> /Users/kosaki/ruby/ruby-193
0x100362000 - 0x100363fff +encdb.bundle ??? (???) <04EBD9DF-B44E-2812-7F4F-CD5B4FB7D63E>
/Users/kosaki/ruby/.ext/x86_64-darwin10.7.4/enc/encdb.bundle

0x100366000 - 0x100367fff +transdb.bundle ??? (???) <73AC857E-8D91-621B-7AAD-9B7039388736> /Users/kosaki/ruby/.ext/x86_64-darwin10.7.4/enc/trans/transdb.bundle
0x7fff5fc00000 - 0x7fff5fc3bdef dyld 132.1 (???) /usr/lib/dyld
0x7fff805a7000 - 0x7fff8065dfff libobjc.A.dylib 227.0.0 (compatibility 1.0.0) <1960E662-D35C-5D98-EB16-D43166AE6A22> /usr/lib/libobjc.A.dylib
0x7fff862ec000 - 0x7fff862f0ff7 libmathCommon.A.dylib 315.0.0 (compatibility 1.0.0) <95718673-FEEE-B6ED-B127-BCDBDB60D4E5> /usr/lib/system/libmathCommon.A.dylib
0x7fff86677000 - 0x7fff866f4fef libstdc++.6.dylib 7.9.0 (compatibility 7.0.0) <35ECA411-2C08-FD7D-11B1-1B7A04921A5C> /usr/lib/libstdc++.6.dylib
0x7fff884d3000 - 0x7fff88694fff libSystem.B.dylib 125.2.10 (compatibility 1.0.0) /usr/lib/libSystem.B.dylib
0x7fff88789000 - 0x7fff887d5fff libauto.dylib ??? (???) <328CCF97-091D-C529-E576-C78583445711> /usr/lib/libauto.dylib
0x7fffffe00000 - 0x7fffffe01fff libSystem.B.dylib ??? (???) /usr/lib/libSystem.B.dylib

Related issues:

Related to Ruby - Bug #4389: "pointer being freed was not allocated" error af...	**Closed**	**02/11/2011**

Associated revisions

Revision 1fdbe0f4 - 06/09/2011 02:45 PM - nagachika (Tomoyuki Chikanaga)

- gc.c (rb_objspace_call_finalizer): use rb_typeddata_is_kind_of() for type check to get rid of a double free when main Thread has singleton class. [ruby-core:36741] [Bug #4828]
- thread.c (rb_obj_is_mutex): add a new utility function.
- vm.c (rb_obj_is_thread): ditto.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@31968 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 06/05/2011 02:51 PM - kosaki (Motohiro KOSAKI)

Hm,

I've spent my time for printf() debugging. and I've found thread_free() is called twice, 1) from run_final() 2) from ruby_vm_destruct().
It seems silly.

#2 - 06/05/2011 03:16 PM - kosaki (Motohiro KOSAKI)

Hm, Linux crash too.

#3 - 06/05/2011 03:45 PM - kosaki (Motohiro KOSAKI)

git bisect has been finished.
I hope every developers confirm test-all result *brefore* commit.

d22130922e7842226d38d59680e4bbb48a28a5f0 is the first bad commit
commit d22130922e7842226d38d59680e4bbb48a28a5f0
Author: ryan ryan@b2dd03c8-39d4-4d8f-98ff-823fe69b080e
Date: Wed Jun 1 03:45:05 2011 +0000

```
Import rubygems 1.8.5 (released @ 137c80f)

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@31885 b2dd03c8-39d4-4d8f-98ff-823fe69b080e
```

:100644 100644 003d346cc2cc01e53d8eb1becf60a6d8b708e104 74535f38277c19c59bbaa75ea2c6584ab55e0076 M ChangeLog
:040000 040000 7d71c2109621efe44b0df4e93f8c55d1e559c7b5 f654e740c4a8a7ae4b70d701ed0303d89d525819 M lib
:040000 040000 ea8124782db9230853610a8bd939b344125ba132 a372090c1faf4df8a51cd2988356e2490aa346b6 M test

#4 - 06/06/2011 12:41 AM - nagachika (Tomoyuki Chikanaga)

hi,

following patch suppress the error.

diff --git a/gc.c b/gc.c
index 28fa233..3499b02 100644
--- a/gc.c
+++ b/gc.c
@@ -3005,7 +3005,7 @@ rb_objspace_call_finalizer(rb_objspace_t *objspace)
while (p < pend) {
if (BUILTIN_TYPE(p) == T_DATA &&
DATA_PTR(p) && RANY(p)->as.data.dfree &&

- [redacted]

- [redacted]

```
p->as.free.flags = 0;
if (RTYPEDDATA_P(p)) {
    RDATA(p)->dfree = RANY(p)->as.typeddata.type->function.dfree;
```

#5 - 06/08/2011 11:11 PM - nagachika (Tomoyuki Chikanaga)

- File typeddata_check_at_finalize.patch added

I refined my patch to use rb_typeddata_is_kind_of() to check Thread and Mutex object,
because rb_obj_is_kind_of() might raise exception and use of rb_typeddata_is_kind_of() seems more efficient.
I'll check in it later.

But I'm hesitating to close this ticket. Is this patch fixes root problem of this issue?
I suspect it is just a makeshift fix.
Shouldn't the main thread object be given special treatment in GC?
Is there any idea?

#6 - 06/08/2011 11:25 PM - naruse (Yui NARUSE)

wow, the patch fixes warnings on valgrind ruby -e'class << Thread.current; end'

#7 - 06/08/2011 11:29 PM - nobu (Nobuyoshi Nakada)

- ruby -v changed from ruby 1.9.3dev (2011-06-04 trunk 31919) [x86_64-darwin10.7.4] to -

Hi,

At Wed, 8 Jun 2011 23:11:29 +0900,
Tomoyuki Chikanaga wrote in [ruby-core:36849]:

> But I'm hesitating to close this ticket. Is this patch fixes root problem of this issue?
> I suspect it is just a makeshift fix.

At least, it fixes the problem, along the current
implementation.

> Shouldn't the main thread object be given special treatment in GC?

I don't think it is enough only for the main thread.

--
Nobu Nakada

#8 - 06/09/2011 11:45 PM - nagachika (Tomoyuki Chikanaga)

- Status changed from Open to Closed

- % Done changed from 0 to 100

This issue was solved with changeset r31968.
Motohiro, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

- gc.c (rb_objspace_call_finalizer): use rb_typeddata_is_kind_of() for type check to get rid of a double free when main Thread has singleton class. [ruby-core:36741] [Bug #4828]
- thread.c (rb_obj_is_mutex): add a new utility function.
- vm.c (rb_obj_is_thread): ditto.

Files

typeddata_check_at_finalize.patch	1.59 KB	06/08/2011	nagachika (Tomoyuki Chikanaga)