Ruby - Bug #3523

win32 exception c0000029 on exit using fibers

07/02/2010 10:42 PM - spatulasnout (B Kelly)

Status:	Third Party's Issue		
Priority:	Normal		
Assignee:	usa (Usaku NAKAMURA)		
Target version:	1.9.2		
ruby -v:	ruby 1.9.2dev (2010-07-01) [i386-mswin32_100]	**Backport:**	

Description

=begin
Hi,

I'm encountering a windows exception c0000029 crash when my application (which uses fibers) runs on v1.9.2. The program runs to completion, and its tests all pass, but the exception occurs when ruby exits.

So far the crash happens consistently on v1.9.2-dev, but never happens on trunk (1.9.3-dev).

I will attempt to isolate a small bit of code which reproduces the problem. But I wanted to post in the meantime in case anyone might have ideas about what could cause this exception, or why it might differ between 1.9.2 and trunk.

Thanks,

Regards,

Bill
=end

History

#1 - 07/12/2010 07:55 PM - mame (Yusuke Endoh)

- Priority changed from Normal to 3

=begin
Hi,

2010/7/2 B Kelly redmine@ruby-lang.org:

> I'm encountering a windows exception c0000029 crash when my application (which uses fibers) runs on v1.9.2. ?The program runs to completion, and its tests all pass, but the exception occurs when ruby exits.
>
> So far the crash happens consistently on v1.9.2-dev, but never happens on trunk (1.9.3-dev).
>
> I will attempt to isolate a small bit of code which reproduces the problem. ?But I wanted to post in the meantime in case anyone might have ideas about what could cause this exception, or why it might differ between 1.9.2 and trunk.

What's going on?

Indeed, this looks like core's bug. But we can't do anything unless
there is reproducible code. Meanwhile, I set the priority to Low.
If you could create code that you can disclose, please reply to this
thread.

--
Yusuke Endoh mame@tsg.ne.jp
=end

#2 - 07/13/2010 04:09 AM - spatulasnout (B Kelly)

=begin
Hi,

Yusuke Endoh wrote:

> What's going on?
>
> Indeed, this looks like core's bug. But we can't do anything unless there is reproducible code. Meanwhile, I set the priority to Low. If you could create code that you can disclose, please reply to this thread.

Sorry for the delay.

Last week I made several attempts to create smaller sample programs which reproduced the error. To my surprise, the smaller programs fail to produce the error, but the larger program still gets the error EVERY time.

Here is what I have learned so far:

1. The 0xc0000029 exception apparently means:
   "An invalid unwind target was encountered during an unwind operation."

2. I suspect the error may have something to do with having a C++ stack frame when switching fibers. I am using EventMachine, and changing fibers on the event callback.

Example:

EM-fiber: callback into ruby (with C++ stack frame)
transfer to Dispatch-fiber
transfer back to EM-fiber
return

But again, I can't reproduce the error with a program as simple as the above.

1. I have back-ported the "cont.c" from trunk to 1.9.2, and the 0xc0000029 problem disappears.

2. Apparently cont.c from trunk uses FIBER_USE_NATIVE, which was briefly in 1.9.2, but was reverted.

That is all I know, so far.

By the way, I was interested to learn the reason why FIBER_USE_NATIVE was reverted in 1.9.2. I think the following discussion may contain the answer, but, Google Translate was not very helpful:

http://redmine.ruby-lang.org/issues/show/3295

If possible, could anyone who understands Japanese please summarize the reason for the revert in English?

Meanwhile, I will make another attempt to reduce the amount of code needed to reproduce the problem, in the next couple days.

Thanks,

Bill

=end

#3 - 07/13/2010 10:02 PM - luislavena (Luis Lavena)

=begin
On Mon, Jul 12, 2010 at 4:09 PM, B Kelly redmine@ruby-lang.org wrote:

> Sorry for the delay.
>
> Last week I made several attempts to create smaller sample programs which reproduced the error. To my surprise, the

> smaller programs fail to produce the error, but the larger
> program still gets the error EVERY time.
>
> Here is what I have learned so far:
>
> 1. The 0xc0000029 exception apparently means:
>   "An invalid unwind target was encountered during an unwind
>   operation."
>
> 2. I suspect the error may have something to do with having
>   a C++ stack frame when switching fibers. I am using
>   EventMachine, and changing fibers on the event callback.

Is EventMachine compiled with the same Visual C?

I can see EventMachine 0.12.10 compiled for Visual C 6.0, so if you haven't forced it, it should not install that version and instead compile it for you.

EventMachine binary gem might have been compiled with MinGW, Since it uses C++, unwind stacks are different and could be the root of the failure.

--
Luis Lavena
AREA 17

Perfection in design is achieved not when there is nothing more to add,
but rather when there is nothing more to take away.
Antoine de Saint-Exupéry

=end

#4 - 07/13/2010 10:22 PM - mame (Yusuke Endoh)

=begin
Hi,

2010/7/13 B Kelly redmine@ruby-lang.org:

> By the way, I was interested to learn the reason why
> FIBER_USE_NATIVE was reverted in 1.9.2. ?I think the
> following discussion may contain the answer, but,
> Google Translate was not very helpful:
>
> http://redmine.ruby-lang.org/issues/show/3295
>
> If possible, could anyone who understands Japanese
> please summarize the reason for the revert in
> English?

FIBER_USE_NATIVE patch aims to improve performance of fiber by using platform-specific fiber feature (such as win32 fiber, getcontext, etc) if available.
But the approach requires many experiences in various platform.
Actually, it caused assert error on Ubuntu, so we determined the patch was too premature for 1.9.2.

> Meanwhile, I will make another attempt to reduce
> the amount of code needed to reproduce the problem,
> in the next couple days.

Good luck!

--
Yusuke Endoh mame@tsg.ne.jp

=end

#5 - 07/14/2010 07:16 AM - spatulasnout (B Kelly)

=begin
Hi,

Luis Lavena wrote:

> Is EventMachine compiled with the same Visual C?

Yes, all extensions are compiled locally, in this
case with cl.exe from Visual Studio 2010 (version
16.00.30319.01).

EventMachine was pulled from git on 2010-06-16

Regards,

Bill

=end

#6 - 07/15/2010 09:45 PM - spatulasnout (B Kelly)

- File test_em_fiber4f.rb added

=begin
Hi,

I've attached a program (in redmine) which reproduces
the c0000029 crash on ruby_1_9_2 / Windows 7.

The program is a single source file, but it does
depend on the EventMachine extension.

The crash happens every time for me. There are
comments at the top of the source file explaining how
EventMachine was built, and describing lines which
can be added or removed to alter or prevent the
c0000029 from occurring.

I'm using Visual Studio 2010 to compile both ruby
and EventMachine.

If there's any way I can help please let me know.

Thanks,

Bill

=end

#7 - 07/16/2010 10:25 AM - usa (Usaku NAKAMURA)

- Status changed from Open to Assigned

- Assignee set to usa (Usaku NAKAMURA)

=begin
I'm tracking this ticket now.
Thank you, Bill! (and of course, Luis and Endoh-san!)

Now I'm doubting that the occasion is that SEH frame is not replaced
when the stack is replaced with fiber switch.
The implementation of Thread of 1.8 has this replacement, but it
seems to have been forgotten in Fiber of 1.9.2...

=end

#8 - 07/16/2010 12:57 PM - usa (Usaku NAKAMURA)

=begin
Hello,

> http://redmine.ruby-lang.org/issues/show/3523

Interim report:

There is a patch for ruby_1_9_2, to save and restore SEH frame.
After this patch is applied, Bill's 0xC0000029 error vanishes!

... But another error 0xC0000374 occurs.
This error means heap corruption.
I cannot confirm yet whether this error is due to ruby or EventMachine.

Index: cont.c

--- cont.c (revision 28653)
+++ cont.c (working copy)
@@ -39,6 +39,9 @@ typedef struct rb_context_struct {
VALUE *machine_register_stack_src;
int machine_register_stack_size;
#endif
+#if _WIN32 && (defined(_M_IX86) || defined(**i386**))

- VALUE win32_exception_list;
  +#endif
  rb_thread_t saved_thread;
  rb_jmpbuf_t jmpbuf;
  size_t machine_stack_size;
  @@ -224,6 +227,44 @@ fiber_memsize(const void *ptr)
  return 0;
  }

+#if _WIN32 && (defined(_M_IX86) || defined(**i386**))
+static inline VALUE
+win32_get_exception_list(void)
+{

- VALUE p;
  +# if defined _MSC_VER
  +# ifdef _M_IX86
  +# if _MSC_VER >= 1310

- [redacted]

+# pragma warning(disable: 4733)
+# endif

- __asm mov eax, fs:[0];
- __asm mov p, eax;
  +# endif
  +# elif defined **GNUC**
  +# ifdef **i386**
- **asm**("movl %%fs:0,%0" : "=r"(p));
  +# endif
  +# endif
- return p;
  +}

+static inline void
+win32_set_exception_list(VALUE p)
+{
+# if defined _MSC_VER
+# ifdef _M_IX86

- __asm mov eax, p;
- __asm mov fs:[0], eax;
  +# endif
  +# elif defined **GNUC**
  +# ifdef **i386**
- **asm**("movl %0,%%fs:0" :: "r"(p));
  +# endif

+# endif
+}
+#endif

static void
cont_save_machine_stack(rb_thread_t *th, rb_context_t *cont)
{
@@ -267,6 +308,9 @@ cont_save_machine_stack(rb_thread_t *th,

```
  MEMCPY(cont->machine_register_stack, cont->machine_register_stack_src, VALUE, size);
```

#endif
+#if _WIN32 && (defined(_M_IX86) || defined(**i386**))

- cont->win32_exception_list = win32_get_exception_list();
  +#endif

  sth->machine_stack_start = sth->machine_stack_end = 0;
  #ifdef __ia64
  @@ -403,6 +447,9 @@ cont_restore_1(rb_context_t *cont)
  }
  #endif
  if (cont->machine_stack_src) {
  +#if _WIN32 && (defined(_M_IX86) || defined(**i386**))

- win32_set_exception_list(cont->win32_exception_list);
  +#endif
  FLUSH_REGISTER_WINDOWS;
  MEMCPY(cont->machine_stack_src, cont->machine_stack,
  VALUE, cont->machine_stack_size);

  Regards,
  --
  U.Nakamura usa@garbagecollect.jp

=end

#9 - 07/16/2010 07:50 PM - mame (Yusuke Endoh)

- Priority changed from 3 to Normal

=begin

=end

#10 - 07/17/2010 04:14 AM - spatulasnout (B Kelly)

=begin
Hi,

U.Nakamura wrote:

> Interim report:
>
> There is a patch for ruby_1_9_2, to save and restore SEH frame.
> After this patch is applied, Bill's 0xC0000029 error vanishes!
>
> ... But another error 0xC0000374 occurs.
> This error means heap corruption.
> I cannot confirm yet whether this error is due to ruby or EventMachine.

Thank you. I've applied the patch, and I initially saw
the 0xC0000374 error, too.

Then I installed some Memory Validator software [1] to
try to learn more (which I have a license for, but had
never installed on my new OS).

However, I am not able to reproduce the 0xC0000374
anymore. With or without Memory Validator running, nor
from the shell, nor within Visual Studio's debugger.
The 0xC0000374 seems to have suddenly disappeared, but,

I have not changed or recompiled anything, as far as I know! :(

Are you still seeing the 0xC0000374 on your end?

Regards,

Bill

[1] http://www.softwareverify.com/cpp/memory/index.html

=end

#11 - 07/17/2010 09:56 PM - spatulasnout (B Kelly)

=begin
Bill Kelly wrote:

> However, I am not able to reproduce the 0xC0000374 anymore.

Very strange. I put the test in a loop, i.e.:

def test_em_fiber
GC.stress = true
10.times {do_test_em_fiber}
end

... and now the 0xC0000374 happens again, every time, after the FIRST time through the loop.

(I already had the GC.stress = true, before.)

Currently MemoryValidator isn't helping much. I will try recompiling with --enable-shared . . . .

Regards,

Bill

=end

#12 - 07/19/2010 06:30 PM - spatulasnout (B Kelly)

=begin
Hi,

Bill Kelly wrote:

> ... and now the 0xC0000374 happens again, every time, after the FIRST time through the loop.
>
> (I already had the GC.stress = true, before.)
>
> Currently MemoryValidator isn't helping much. I will try recompiling with --enable-shared . . . .

(Note: --enable-shared didn't change the Makefile produced by extconf.rb at all. So, nevermind --enable-shared.)

When I put the test in a loop, and if I don't get the 0xC0000374, I see the following:

```
test/test_em_fiber4f.rb:151: [BUG] Segmentation fault
ruby 1.9.2dev (2010-07-15) [i386-mswin32_100]

-- control frame ----------
c:0025 p:---- s:0096 b:0096 l:000095 d:000095 CFUNC  :transfer
c:0024 p:0046 s:0093 b:0093 l:0024f0 d:0024f0 METHOD test/test_em_fiber4f.rb:151
c:0023 p:0083 s:0087 b:0087 l:00265c d:00265c METHOD test/test_em_fiber4f.rb:260
c:0022 p:0076 s:0080 b:0080 l:000079 d:000079 METHOD test/test_em_fiber4f.rb:241
c:0021 p:0021 s:0076 b:0076 l:000075 d:000075 METHOD test/test_em_fiber4f.rb:182
c:0020 p:---- s:0072 b:0072 l:000071 d:000071 FINISH
```

```
c:0019 p:---- s:0070 b:0070 l:000069 d:000069 CFUNC  :run_machine
c:0018 p:0248 s:0067 b:0067 l:000066 d:000066 METHOD m:/dev/ruby-build/v1_9_2-pure/lib/ruby/site_ruby/1.9.1/ev
entmachine.rb:195
c:0017 p:0131 s:0060 b:0060 l:001df4 d:001df4 METHOD test/test_em_fiber4f.rb:527
c:0016 p:0009 s:0055 b:0055 l:000047 d:000054 BLOCK  test/test_em_fiber4f.rb:512
c:0015 p:---- s:0053 b:0053 l:000052 d:000052 FINISH
c:0014 p:---- s:0051 b:0051 l:000050 d:000050 CFUNC  :times
c:0013 p:0030 s:0048 b:0048 l:000047 d:000047 METHOD test/test_em_fiber4f.rb:512
c:0012 p:0063 s:0045 b:0045 l:000044 d:000044 METHOD m:/dev/ruby-build/v1_9_2-pure/lib/ruby/1.9.1/minitest/uni
t.rb:680
c:0011 p:0091 s:0039 b:0039 l:000020 d:000038 BLOCK  m:/dev/ruby-build/v1_9_2-pure/lib/ruby/1.9.1/minitest/uni
t.rb:641
c:0010 p:---- s:0034 b:0034 l:000033 d:000033 FINISH
c:0009 p:---- s:0032 b:0032 l:000031 d:000031 CFUNC  :each
c:0008 p:0026 s:0029 b:0029 l:000020 d:000028 BLOCK  m:/dev/ruby-build/v1_9_2-pure/lib/ruby/1.9.1/minitest/uni
t.rb:635
c:0007 p:---- s:0026 b:0026 l:000025 d:000025 FINISH
c:0006 p:---- s:0024 b:0024 l:000023 d:000023 CFUNC  :each
c:0005 p:0082 s:0021 b:0021 l:000020 d:000020 METHOD m:/dev/ruby-build/v1_9_2-pure/lib/ruby/1.9.1/minitest/uni
t.rb:634
c:0004 p:0188 s:0016 b:0016 l:000015 d:000015 METHOD m:/dev/ruby-build/v1_9_2-pure/lib/ruby/1.9.1/minitest/uni
t.rb:594
c:0003 p:0041 s:0007 b:0007 l:00212c d:000006 BLOCK  m:/dev/ruby-build/v1_9_2-pure/lib/ruby/1.9.1/minitest/uni
t.rb:492
c:0002 p:---- s:0004 b:0004 l:000003 d:000003 FINISH
c:0001 p:0000 s:0002 b:0002 l:00060c d:00060c TOP
---------------------------
-- Ruby level backtrace information ----------------------------------------
m:/dev/ruby-build/v1_9_2-pure/lib/ruby/1.9.1/minitest/unit.rb:492:in `block in autorun'
m:/dev/ruby-build/v1_9_2-pure/lib/ruby/1.9.1/minitest/unit.rb:594:in `run'
m:/dev/ruby-build/v1_9_2-pure/lib/ruby/1.9.1/minitest/unit.rb:634:in `run_test_suites'
m:/dev/ruby-build/v1_9_2-pure/lib/ruby/1.9.1/minitest/unit.rb:634:in `each'
m:/dev/ruby-build/v1_9_2-pure/lib/ruby/1.9.1/minitest/unit.rb:635:in `block in run_test_suites'
m:/dev/ruby-build/v1_9_2-pure/lib/ruby/1.9.1/minitest/unit.rb:635:in `each'
m:/dev/ruby-build/v1_9_2-pure/lib/ruby/1.9.1/minitest/unit.rb:641:in `block (2 levels) in run_test_suites'
m:/dev/ruby-build/v1_9_2-pure/lib/ruby/1.9.1/minitest/unit.rb:680:in `run'
test/test_em_fiber4f.rb:512:in `test_em_fiber'
test/test_em_fiber4f.rb:512:in `times'
test/test_em_fiber4f.rb:512:in `block in test_em_fiber'
test/test_em_fiber4f.rb:527:in `do_test_em_fiber'
m:/dev/ruby-build/v1_9_2-pure/lib/ruby/site_ruby/1.9.1/eventmachine.rb:195:in `run'
m:/dev/ruby-build/v1_9_2-pure/lib/ruby/site_ruby/1.9.1/eventmachine.rb:195:in `run_machine'
test/test_em_fiber4f.rb:182:in `receive_data'
test/test_em_fiber4f.rb:241:in `rpc_connection_established'
test/test_em_fiber4f.rb:260:in `rpc_dispatch'
test/test_em_fiber4f.rb:151:in `run_in_fiber'
test/test_em_fiber4f.rb:151:in `transfer'

[NOTE]
You may have encountered a bug in the Ruby interpreter or extension libraries.
Bug reports are welcome.
For details: http://www.ruby-lang.org/bugreport.html
```

Note, I get no such crash on 1.9.2 if I back-port cont.c from trunk.
(I can loop the test hundreds of times with no crash.)

So I can't prove it's not EventMachine's fault, but... It seems to
have to do with cont.c.

Regards,

Bill

=end

#13 - 07/20/2010 03:12 PM - usa (Usaku NAKAMURA)

=begin
Hello,

In message "[ruby-core:31351] Re: [Bug #3523][Assigned] win32 exception c0000029 on exit using fibers"
on Jul.19,2010 18:30:23, billk@cts.com wrote:

> When I put the test in a loop, and if I don't get the
> 0xC0000374, I see the following:

```
test/test_em_fiber4f.rb:151: [BUG] Segmentation fault
ruby 1.9.2dev (2010-07-15) [i386-mswin32_100]
```

Great!
Now I can pay attention only to implementation of transfer
method.

Thus, I have not found the cause yet.
From the phenomenon and the point of the problem, it seems that
restoring the stack is a front runner...

Regards,

U.Nakamura usa@garbagecollect.jp

=end

#14 - 07/21/2010 06:31 PM - usa (Usaku NAKAMURA)

=begin
Hello,

In message "[ruby-core:31366] Re: [Bug #3523][Assigned] win32 exception c0000029 on exit using fibers"
on Jul.20,2010 15:11:56, usa@garbagecollect.jp wrote:

> Thus, I have not found the cause yet.

But I've found a workaround for you, Bill.

In definition of ZZZZ::PROTO::RPCMessageDispatcher#rpc_connection_unbind,
you should use
@msg_portmap[MSGPORT_ROOT].call(true, :rpc_unbind)
instead of
rpc_dispatch(MSGPORT_ROOT, MSGPORT_ROOT, :rpc_unbind)
.

Regards,

U.Nakamura usa@garbagecollect.jp

=end

#15 - 07/21/2010 09:41 PM - spatulasnout (B Kelly)

=begin
Hi,

U.Nakamura wrote:

> But I've found a workaround for you, Bill.
>
> In definition of ZZZZ::PROTO::RPCMessageDispatcher#rpc_connection_unbind,
> you should use
> @msg_portmap[MSGPORT_ROOT].call(true, :rpc_unbind)
> instead of
> rpc_dispatch(MSGPORT_ROOT, MSGPORT_ROOT, :rpc_unbind)
> .

Interesting.

Do we know why it is unsafe to use run_in_fiber specifically during the
unbind callback?

Note that test_em_fiber4f.rb represents an extremely simplified version
of the system, and that normally rpc_dispatch is also frequently called
from another EventMachine callback: receive_data.

So I am wondering if we have a reason to believe that it is *only* the
unbind callback that is unsafe? And if we should expect the
receive_data callback to be safe for run_in_fiber, unlike unbind?

(Unfortunately, unlike c0000029, I'm not yet able to reproduce the current memory corruption crash when running tests that exercise the real system. So I don't know if this change in test_em_fiber4f.rb makes a difference in the real system or not, since I can't get the real system to crash either way, yet.)

Thanks,

Bill

=end

#16 - 07/22/2010 09:44 AM - usa (Usaku NAKAMURA)

=begin
Hello,

In message "[ruby-core:31403] Re: [Bug #3523][Assigned] win32 exception c0000029 on exit using fibers"
on Jul.21,2010 21:41:45, billk@cts.com wrote:

> Interesting.
>
> Do we know why it is unsafe to use run_in_fiber specifically during the
> unbind callback?

Because it's called from the C++ destructor.
When swiching fibers in ruby code which is called from C++ destructor, the heap of destructing C++ object seems to be corrupted.
I've not understood yet why switching fibers corrupt the heap and how to avoid the corruption.

So, now I can say only that you should not switch fibers with the posibility to be called in C++ destructor.

Regards,

U.Nakamura usa@garbagecollect.jp

=end

#17 - 07/22/2010 12:29 PM - spatulasnout (B Kelly)

=begin
U.Nakamura wrote:

> So, now I can say only that you should not switch fibers
> with the posibility to be called in C++ destructor.

Ah. Thank you for the clarification!

Regards,

Bill

=end

#18 - 07/29/2010 09:29 PM - mame (Yusuke Endoh)

- Priority changed from Normal to 3

=begin
Hi,

I'm changing this to Low priority.
I'm sorry because you did good work, though.

According to unak's (painful) investigation, this bug seems to be caused by resuming fiber from C++ destructor.

It is very horrible for me. I can't imagine what it will cause.
It is good to ask the author of eventmachine to investigate this,

though this may be actually Ruby core's bug. In principle, Ruby
assumes that extension library be written in C.

In addition, I believe this is a rare case.

--
Yusuke Endoh mame@tsg.ne.jp
=end

#19 - 07/30/2010 08:11 AM - spatulasnout (B Kelly)

=begin
Hi,

Yusuke Endoh wrote:

> I'm changing this to Low priority.
> I'm sorry because you did good work, though.
>
> According to unak's (painful) investigation, this bug seems to be
> caused by resuming fiber from C++ destructor.
>
> It is very horrible for me. I can't imagine what it will cause.
> It is good to ask the author of eventmachine to investigate this,
> though this may be actually Ruby core's bug. In principle, Ruby
> assumes that extension library be written in C.
>
> In addition, I believe this is a rare case.

Thanks for the update. I've conveyed the information to the
EventMachine list.

By the way, I am still using 1.9.2 with cont.d back-ported
from trunk, with FIBER_USE_NATIVE enabled.

So far it has been flawless. I haven't been able to reproduce
any of these errors when trunk's cont.c is used.

Regards,

Bill

=end

#20 - 06/11/2011 02:40 PM - ko1 (Koichi Sasada)

Can we close this issue?

#21 - 10/31/2012 04:27 PM - usa (Usaku NAKAMURA)

- Status changed from Assigned to Third Party's Issue

Although it is too late, I think that we don't handle this now,
so change this ticket as TPI.

Files

test_em_fiber4f.rb	14.2 KB	07/15/2010	spatulasnout (B Kelly)