Deadlock when calling Thread#join from signal interrupt context

05/09/2012 06:13 AM - ender672 (Timothy Elliott)

Status: Closed
Priority: Normal
Assignee: kosaki (Motohiro KOSAKI)
Target version: 2.0.0

Description

The interpreter can deadlock when calling Thread#join both from the main context and from the signal handler context.

t = Thread.new{ sleep 3 }

Signal.trap "SIGINT" do
t.join
end

puts 'Press ctrl + c now'
t.join

The above will deadlock on linux x86_64 with ruby 1.9.x and ruby trunk. It works fine with ruby 1.8.7-p352 and JRuby.

Associated revisions

Revision 37852 - 11/26/2012 08:05 AM - kosaki (Motohiro KOSAKI)

- vm_core.h (rb_thread_struct): added 'in_trap' member for marking running trap handler.
- signal.c (signal_exec): turn on in_trap when running trap.
- thread.c (Init_Thread, thread_create_core): initialize in_trap when creating new threads.
- thread.c (thread_join_m): raise ThreadError when running trap handler.Bug [#6416][ruby-core:44956]
- test/ruby/test_thread.rb (test_thread_join_in_trap): new test for the above.

Revision 37852 - 11/26/2012 08:05 AM - kosaki (Motohiro KOSAKI)

- vm_core.h (rb_thread_struct): added 'in_trap' member for marking running trap handler.
- signal.c (signal_exec): turn on in_trap when running trap.
- thread.c (Init_Thread, thread_create_core): initialize in_trap when creating new threads.
- thread.c (thread_join_m): raise ThreadError when running trap handler.Bug [#6416][ruby-core:44956]
- test/rfc/dump_thread.rb (test_thread_join_in_trap): new test for the above.

Revision 37852 - 11/26/2012 08:05 AM - kosaki (Motohiro KOSAKI)

- vm_core.h (rb_thread_struct): added 'in_trap' member for marking running trap handler.
- signal.c (signal_exec): turn on in_trap when running trap.
- thread.c (Init_Thread, thread_create_core): initialize in_trap when creating new threads.
- thread.c (thread_join_m): raise ThreadError when running trap handler.Bug [#6416][ruby-core:44956]
- test/rfc/dump_thread.rb (test_thread_join_in_trap): new test for the above.

Revision 37852 - 11/26/2012 08:05 AM - kosaki (Motohiro KOSAKI)

- vm_core.h (rb_thread_struct): added 'in_trap' member for marking running trap handler.
- signal.c (signal_exec): turn on in_trap when running trap.
- thread.c (Init_Thread, thread_create_core): initialize in_trap when creating new threads.
- thread.c (thread_join_m): raise ThreadError when running trap handler.Bug [#6416][ruby-core:44956]
- test/rfc/dump_thread.rb (test_thread_join_in_trap): new test for the above.
- vm_core.h (rb_thread_struct): added 'in_trap' member for marking running trap handler.
- signal.c (signal_exec): turn on in_trap when running trap.
- thread.c (Init_Thread, thread_create_core): initialize in_trap when creating new threads.
- thread.c (thread_join_m): raise ThreadError when running trap handler. Bug [#6416][ruby-core:44956]
- test/ruby/test_thread.rb (test_thread_join_in_trap): new test for the above.

Revision 37852 - 11/26/2012 08:05 AM - kosaki (Motohiro KOSAKI)

- vm_core.h (rb_thread_struct): added 'in_trap' member for marking running trap handler.
- signal.c (signal_exec): turn on in_trap when running trap.
- thread.c (Init_Thread, thread_create_core): initialize in_trap when creating new threads.
- thread.c (thread_join_m): raise ThreadError when running trap handler. Bug [#6416][ruby-core:44956]
- test/ruby/test_thread.rb (test_thread_join_in_trap): new test for the above.

History

#1 - 05/17/2012 12:32 AM - mame (Yusuke Endoh)
  - Status changed from Open to Assigned
  - Assignee set to kosaki (Motohiro KOSAKI)

Kosaki-san, could you take a look?

--
Yusuke Endoh mame@tsg.ne.jp

#2 - 05/18/2012 05:04 PM - kosaki (Motohiro KOSAKI)
  - Assignee changed from kosaki (Motohiro KOSAKI) to ko1 (Koichi Sasada)

Hi ko1,

This is because thread_join() is not designed reentrant. It can insert a thread twice join_list. And then, join_list is going to become circular ring and it lead to deadlock.

So, I think th.join in trap context should raise ThreadError. What do you think?

#3 - 10/30/2012 09:13 AM - ko1 (Koichi Sasada)
  - Target version set to 2.0.0

Please give me a time.

#4 - 11/26/2012 09:35 AM - ko1 (Koichi Sasada)
  - Assignee changed from ko1 (Koichi Sasada) to kosaki (Motohiro KOSAKI)

I agree with kosaki-san's comment.

#5 - 11/26/2012 08:02 PM - kosaki (Motohiro KOSAKI)
  - Status changed from Assigned to Closed

Fixed at r37852.
Thanks, Timothy!

#6 - 11/28/2012 11:31 AM - authorNari (Narihiro Nakamura)

Hello

I need sometimes to call Thread#join in Signal.trap when I want to implement safe termination in a server. For instance: https://gist.github.com/4158509

Is it a wrong use case in the first place?

Thanks.

#7 - 11/28/2012 11:46 AM - kosaki (Motohiro KOSAKI)

Nari,
In your case, main thread and trap handler uses the same mutex and it is racy and deadlockable. Even if I revert my change, it wouldn't work.

kosaki (Motohiro KOSAKI) wrote:

Nari,

In your case, main thread and trap handler uses the same mutex and it is racy and deadlockable. Even if I revert my change, it wouldn't work.

I see, Thanks!