Ruby trunk - Bug #6575

Thread#kill sets rb_errinfo() to Fixnum 8 after rb_protect(function, data, &error_tag)

06/11/2012 08:27 PM - ibc (Iñaki Baz Castillo)

Status:	Rejected		
Priority:	Normal		
Assignee:	ko1 (Koichi Sasada)		
Target version:	1.9.3		
ruby -v:	ruby 1.9.3p0 (2011-10-30 revision 33570) [x86_64-linux]	**Backport:**	

Description

Using rb_protect() I've realized that when the thread is killed by other thread using Thread#kill, the error_tag passed to rb_protect() is set to 8 and rb_errinfo() returns Fixnum 8:

ret = rb_protect(function, data, &error_tag);

// While executing function() in Ruby land, our thread is
// killed by Thread.kill.

// If now I inspect rb_errinfo() it returns Fixnum 8, and
// error_tag it's set to integer 8.

Is it the expected behaviour? Wouldn't make more sense rb_errinfo() to return some kind of a new exception ThreadKilledException or whatever instead of Fixnum 8?

Related issues:

Related to Ruby trunk - Bug #5993: Thread.new{ Fiber.new { Thread.exit }.resu...	**Closed**	**02/10/2012**

History

#1 - 06/13/2012 07:01 AM - ibc (Iñaki Baz Castillo)

Hi, if this is the expected behavior I would really appreciate a confirmation :)

#2 - 06/13/2012 11:53 AM - ko1 (Koichi Sasada)

Hi,

(2012/06/11 20:27), ibc (Iñaki Baz Castillo) wrote:

> Using rb_protect() I've realized that when the thread is killed by other thread using Thread#kill, the error_tag passed to rb_protect() is set to 8 and rb_errinfo() returns Fixnum 8:
>
> ret = rb_protect(function, data, &error_tag);
>
> // While executing function() in Ruby land, our thread is
> // killed by Thread.kill.
>
> // If now I inspect rb_errinfo() it returns Fixnum 8, and
> // error_tag it's set to integer 8.
>
> Is it the expected behaviour? Wouldn't make more sense rb_errinfo() to return some kind of a new exception ThreadKilledException or whatever instead of Fixnum 8?
>
> I don't have an idea.

Could you show us the complete *small* example on it?
I want to try on my environment.

Thanks,
Koichi

--
// SASADA Koichi at atdot dot net

#3 - 06/13/2012 01:33 PM - nagachika (Tomoyuki Chikanaga)

Hi,

Just for reference. r35622 could be related, or ticket #5993 seems related issue.
ibc-san, could you try to reproduce the problem on trunk?

Thanks,

#4 - 06/13/2012 09:29 PM - ibc (Iñaki Baz Castillo)

Hi, I've installed rvm and ruby-head, which is retrieved from https://github.com/ruby/ruby/. Unfortunately the patch you mean, which is this:

https://github.com/ruby/ruby/commit/38d3b013b7733d9ccd66c011d74c00b35bb704c4

was reverted:

https://github.com/ruby/ruby/commit/cc08e95b206f8c98af9509f99339f3c8655481e7

so...

#5 - 06/13/2012 09:47 PM - ibc (Iñaki Baz Castillo)

BTW, is Ruby-head (ruby 2.0.0dev (2012-06-13 trunk 36062) [x86_64-linux]) stable enough?

Running my C extension (which uses blocking region for running a libuv loop) with 1.9.3-p0 works perfectly, but using 2.0.0dev I get assertion errors due to states that should never happen in my C code. Are there important changes in the blocking-region stuff in 2.0.0?

#6 - 06/14/2012 10:26 AM - naruse (Yui NARUSE)

- Status changed from Open to Feedback

nagachika (Tomoyuki Chikanaga) wrote:

> Hi,
>
> Just for reference. r35622 could be related, or ticket #5993 seems related issue.
> ibc-san, could you try to reproduce the problem on trunk?

The correct fix of it is r35625.
The commit message maybe helps your problem.

#7 - 06/14/2012 11:59 PM - ibc (Iñaki Baz Castillo)

Thanks, that's more or less the workaround I've applied in my code: if rb_protect() detects an error and rb_errinfo() returns a Fixnum, then I don't raise it, but instead set the error to Interrupt and then raise it.

#8 - 11/06/2012 08:32 PM - mame (Yusuke Endoh)

- Status changed from Feedback to Rejected

- Assignee set to ko1 (Koichi Sasada)

This is not a bug. So I'm closing this ticket.

Because Thread#kill should not be rescue'd so easily, it does throw a special exception which has no class, to make it hard to rescue.

Incidentally, the value 8 means INT2FIX(TAG_FATAL) which is set in rb_threadptr_to_kill:

```
1690 static void
1691 rb_threadptr_to_kill(rb_thread_t *th)
1692 {
1693     rb_threadptr_async_errinfo_clear(th);
1694     th->status = THREAD_TO_KILL;
1695     th->errinfo = INT2FIX(TAG_FATAL);
1696     TH_JUMP_TAG(th, TAG_FATAL);
1697 }
```

If you want to know the detailed rationale, please ask ko1 in the mailing list.
If you wish, please open another feature request with the concrete motivation.

--
Yusuke Endoh mame@tsg.ne.jp