Ruby - Bug #17494

ruby is hanged when using activesupport + rspec + rspec-parameterized

12/31/2020 11:47 AM - sue445 (Go Sueyoshi)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	ruby 3.0.0p0	**Backport:**	2.6: DONTNEED, 2.7: DONTNEED, 3.0: DONE

Description

Example code

Gemfile

```
# frozen_string_literal: true

source "https://rubygems.org"

git_source(:github) {|repo_name| "https://github.com/#{repo_name}" }

gem "activesupport", "6.1.0"
gem "rspec", "3.10.0"
gem "rspec-parameterized", "0.4.2"
```

spec file

```
require "active_support/all"
require "rspec-parameterized"

describe "CLI" do
  subject do
    # Expected error, but actual hunged here
    cli.foo # <- hunged here
  end

  it { expect { subject }.to raise_error }
end

xdescribe "GitlabMrRelease::Project" do
  describe "#api_version" do
    using RSpec::Parameterized::TableSyntax

    where(:api_endpoint, :expected) do
      "http://example.com/api/v4/" | 4
    end

    with_them do
      # it { should eq expected }
    end
  end
end
```

all codes are here.

https://github.com/sue445/ruby_3_0_0_bug_report_20201231

Expected

spec is successful (This is the behavior up to ruby 2.7.2)

Actual

hunged at line 7

History

#1 - 12/31/2020 12:04 PM - sue445 (Go Sueyoshi)

- Subject changed from ruby is hunged when using activesupport + rspec + rspec-parameterized to ruby is hanged when using activesupport + rspec + rspec-parameterized

#2 - 01/01/2021 07:34 AM - yahonda (Yasuo Honda)

Looks like this behavior has been triggered by https://github.com/ruby/ruby/commit/b9007b6c548f91e88fd3f2ffa23de740431fa969

- Created one file repro named rep17494.rb by adding require "rspec/autorun"

```
require "bundler/inline"

gemfile(true) do
  source "https://rubygems.org"

  git_source(:github) {|repo_name| "https://github.com/#{repo_name}" }

  gem "activesupport", "6.1.0"
  gem "rspec", "3.10.0"
  gem "rspec-parameterized", "0.4.2"
end

require "rspec/autorun"
require "active_support/all"
require "rspec-parameterized"

describe "CLI" do
  subject do
    # Expected error, but actual hunged here
    cli.foo # <- hunged here
  end

  it { expect { subject }.to raise_error }
end

describe "GitlabMrRelease::Project" do
  describe "#api_version" do
    using RSpec::Parameterized::TableSyntax

    where(:api_endpoint, :expected) do
      "http://example.com/api/v4/" | 4
    end

    with_them do
      # it { should eq expected }
    end
  end
end
```

- Execute rep17494.rb script against Ruby as of f2286925f08406bc857f7b03ad6779a5d61443ae which is the parent commit of b9007b6c548f91e88fd3f2ffa23de740431fa969

```
$ docker run -it --rm -v "$PWD":/usr/src/myapp -w /usr/src/myapp rubylang/rubyfarm:f2286925f08406bc857f7b03ad6
779a5d61443ae ruby rep17494.rb
```

It works as expected 1 example, 0 failures.

- Execute rep17494.rb script against Ruby as of b9007b6c548f91e88fd3f2ffa23de740431fa969

```
$ docker run -it --rm -v "$PWD":/usr/src/myapp -w /usr/src/myapp rubylang/rubyfarm:b9007b6c548f91e88fd3f2ffa23
de740431fa969 ruby rep17494.rb
```

It hangs.

Of course, this reproduce case does not require Docker if you can build Ruby as of each commit.

#3 - 01/22/2021 05:41 AM - sue445 (Go Sueyoshi)

Workaround

require only minimal files.

In this case, stop to require "active_support/all"

e.g.

```
# require "active_support/all"
require "active_support/core_ext/time/zones"
require "active_support/core_ext/array/wrap"
require "active_support/core_ext/hash/keys"
require "active_support/core_ext/time/calculations"
```

#4 - 01/22/2021 04:32 PM - alpaca-tc (Hiroyuki Ishii)

I investigated this issue deeply based on yhonda's example code. Then I succeeded to create tiny reproduction code.

```
Object.prepend(Module.new)

using(Module.new {
  refine Object do
    def hello; end
  end
})

Object.new.hello

module M
  def hello; end
end

class A
  include M
end

Object.instance_methods #=> hanged!!!
```

#5 - 01/25/2021 12:42 AM - shyouhei (Shyouhei Urabe)

@alpaca-tc Thannk you, I can reproduce that too.

```
zsh % LC_ALL=C gdb --args ./miniruby ~/tmp.rb
GNU gdb (Ubuntu 8.2-0ubuntu1~18.04) 8.2
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./miniruby...done.
(gdb) run
Starting program: /miniruby /tmp.rb
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
# <<<<< Hangs
^C
Program received signal SIGINT, Interrupt.
0x0000555555844e8f in hash_table_index (tbl=0x555555cdfcf0, key=1942) at id_table.c:132
132         if (tbl->capa > 0) {
(gdb) bt
#0  0x0000555555844e8f in hash_table_index (tbl=0x555555cdfcf0, key=1942) at id_table.c:132
#1  0x0000555555844e0c in rb_id_table_lookup (tbl=0x555555cdfcf0, id=31073, valp=0x7fffffff99a0) at id_table.c
:230
#2  0x00005555558b42ae in lookup_method_table (klass=93824999686600, id=31073) at vm_method.c:701
#3  0x00005555558b6dc4 in search_method (klass=93824999686600, id=31073, defined_class_ptr=0x0) at vm_method.c
```

```
:981
#4  0x00005555558b4ff5 in search_method_protect (klass=93824999686544, id=31073, defined_class_ptr=0x0) at vm_
method.c:997
#5  0x00005555558b5f80 in resolve_refined_method (refinements=8, me=0x555555c6ba08, defined_class_ptr=0x0) at
vm_method.c:1267
#6  0x00005555558b5e31 in rb_resolve_refined_method (refinements=8, me=0x555555c6be30) at vm_method.c:1275
#7  0x00005555555cbb56 in method_entry_i (key=31073, value=93824999669296, data=0x7fffffff9bb8) at class.c:137
2
#8  0x00005555558452d8 in rb_id_table_foreach (tbl=0x555555dce520, func=0x5555555cbb00 <method_entry_i>, data=
0x7fffffff9bb8) at id_table.c:299
#9  0x00005555555ce5b4 in add_instance_method_list (mod=93825000677184, me_arg=0x7fffffff9bb8) at class.c:1393
#10 0x00005555555cb040 in class_instance_method_list (argc=0, argv=0x7ffff7ecb048, mod=93825000677184, obj=0,
func=0x5555555cb0e0 <ins_methods_i>) at class.c:1429
#11 0x00005555555caf1c in rb_class_instance_methods (argc=0, argv=0x7ffff7ecb048, mod=93825000677184) at class
.c:1470
#12 0x00005555558e3cea in ractor_safe_call_cfunc_m1 (recv=93825000677184, argc=0, argv=0x7ffff7ecb048, func=0x
5555555caef0 <rb_class_instance_methods>) at vm_insnhelper.c:2706
#13 0x00005555558deba5 in vm_call_cfunc_with_frame (ec=0x555555c567d0, reg_cfp=0x7ffff7fcaf90, calling=0x7ffff
fffa020) at vm_insnhelper.c:2896
#14 0x00005555558d7830 in vm_call_cfunc (ec=0x555555c567d0, reg_cfp=0x7ffff7fcaf90, calling=0x7fffffffa020) at
 vm_insnhelper.c:2917
#15 0x00005555558d71c4 in vm_call_method_each_type (ec=0x555555c567d0, cfp=0x7ffff7fcaf90, calling=0x7fffffffa
020) at vm_insnhelper.c:3386
#16 0x00005555558d6cb2 in vm_call_method (ec=0x555555c567d0, cfp=0x7ffff7fcaf90, calling=0x7fffffffa020) at vm
_insnhelper.c:3479
#17 0x00005555558b9205 in vm_call_general (ec=0x555555c567d0, reg_cfp=0x7ffff7fcaf90, calling=0x7fffffffa020)
at vm_insnhelper.c:3522
#18 0x00005555558cdbc1 in vm_sendish (ec=0x555555c567d0, reg_cfp=0x7ffff7fcaf90, cd=0x555555dcd6d0, block_hand
ler=0, method_explorer=mexp_search_method) at vm_insnhelper.c:4497
#19 0x00005555558a0dbe in vm_exec_core (ec=0x555555c567d0, initial=0) at insns.def:789
#20 0x00005555558c306e in rb_vm_exec (ec=0x555555c567d0, mjit_enable_p=true) at vm.c:2162
#21 0x00005555558c42a0 in rb_iseq_eval_main (iseq=0x555555c708c8) at vm.c:2419
#22 0x0000555555656757 in rb_ec_exec_node (ec=0x555555c567d0, n=0x555555c708c8) at eval.c:317
#23 0x000055555565659c in ruby_run_node (n=0x555555c708c8) at eval.c:375
#24 0x00005555557ce44 in main (argc=2, argv=0x7fffffffc928) at main.c:47
(gdb)
```

Edit: updated the backtrace using -O0 binary.

#6 - 01/25/2021 08:51 AM - nobu (Nobuyoshi Nakada)

This patch seems to avoid the loop for the time being.

```
diff --git i/class.c w/class.c
index a62ae669f84..61f3ece40f4 100644
--- i/class.c
+++ w/class.c
@@ -1369,7 +1369,7 @@ method_entry_i(ID key, VALUE value, void *data)

     if (me->def->type == VM_METHOD_TYPE_REFINED) {
   VALUE owner = me->owner;
- me = rb_resolve_refined_method(Qnil, me);
+ me = rb_method_entry_with_refinements(owner, me->called_id, NULL);
   if (!me) return ID_TABLE_CONTINUE;
   if (!arg->recur && me->owner != owner) return ID_TABLE_CONTINUE;
     }
```

#7 - 01/26/2021 12:45 AM - nobu (Nobuyoshi Nakada)

The previous patch failed an assertion.
https://github.com/nobu/ruby/runs/1760739746?check_suite_focus=true#step:15:288

> | Assertion Failed: ../src/vm_method.c:990:search_method:me == NULL || !METHOD_ENTRY_INVALIDATED(me)

Maybe the real cause is an access to the invalidated method entry?

#8 - 01/26/2021 05:32 AM - yahonda (Yasuo Honda)

This minimum case hangs since https://github.com/ruby/ruby/commit/b9007b6c548f91e88fd3f2ffa23de740431fa969

- Save this minimum case as rep17494_min.rb, added some p method to see if it hangs or not.

```
Object.prepend(Module.new)
```

```
using(Module.new {
  refine Object do
    def hello; end
  end
})

Object.new.hello

module M
  def hello; end
end

class A
  include M
end

p 'Showing Object.instance_methods'
p Object.instance_methods
```

- Run this rep17494_min.rb against as of b9007b6c548f91e88fd3f2ffa23de740431fa969 using rubylang/rubyfarm:b9007b6c548f91e88fd3f2ffa23de740431fa969 docker image

```
% docker run -it --rm -v "$PWD":/usr/src/myapp -w /usr/src/myapp rubylang/rubyfarm:b9007b6c548f91e88fd3f2ffa23
de740431fa969 ruby rep17494_min.rb
"Showing Object.instance_methods"
```

It hangs at p Object.instance_methods.

- Run this rep17494_min.rb against as of f2286925f08406bc857f7b03ad6779a5d61443ae, which is the parent commit of b9007b6c548f91e88fd3f2ffa23de740431fa969

```
% docker run -it --rm -v "$PWD":/usr/src/myapp -w /usr/src/myapp rubylang/rubyfarm:f2286925f08406bc857f7b03ad6
779a5d61443ae ruby rep17494_min.rb

"Showing Object.instance_methods"
[:dup, :itself, :yield_self, :then, :taint, :tainted?, :untaint, :untrust, :untrusted?, :trust, :frozen?, :met
hods, :singleton_methods, :protected_methods, :private_methods, :public_methods, :instance_variables, :instanc
e_variable_get, :instance_variable_set, :instance_variable_defined?, :remove_instance_variable, :instance_of?,
 :kind_of?, :is_a?, :tap, :singleton_class, :clone, :public_send, :method, :public_method, :singleton_method,
:define_singleton_method, :extend, :<=>, :to_enum, :enum_for, :===, :=~, :!~, :nil?, :eql?, :respond_to?, :fre
eze, :inspect, :object_id, :send, :to_s, :display, :class, :hash, :__send__, :!, :__id__, :==, :!=, :equal?, :
instance_eval, :instance_exec]
```

It shows the result of p Object.instance_methods.
These steps do not require Docker if each Ruby build is available locally.

#9 - 03/11/2021 12:01 AM - jeremyevans0 (Jeremy Evans)

One possible workaround for this is a checking for an immediate loop in resolve_refined_method:

```
diff --git a/vm_method.c b/vm_method.c
index 2573e708ba..ebfe686a27 100644
--- a/vm_method.c
+++ b/vm_method.c
@@ -1245,7 +1245,7 @@ resolve_refined_method(VALUE refinements, const rb_method_entry_t *me, VALUE *de
 {
     while (me && me->def->type == VM_METHOD_TYPE_REFINED) {
        VALUE refinement;
-        const rb_method_entry_t *tmp_me;
+        const rb_method_entry_t *tmp_me, *prev_me = me;
         VALUE super;

        refinement = find_refinement(refinements, me->owner);
@@ -1269,6 +1269,9 @@ resolve_refined_method(VALUE refinements, const rb_method_entry_t *me, VALUE *de
         }

         me = search_method_protect(super, me->called_id, defined_class_ptr);
+        if (me == prev_me) {
+            return 0;
+        }
     }
     return me;
 }
```

This fixes the case in the example, but maybe there are other more complex cases that it wouldn't catch. However, even if it won't catch all cases,

until we have solved the underlying issue, this seems like a reasonable thing to add. I can submit a pull request for this if other committers are in favor.

#10 - 07/01/2021 09:44 PM - jeremyevans0 (Jeremy Evans)

- Status changed from Open to Closed

- Backport changed from 2.5: UNKNOWN, 2.6: UNKNOWN, 2.7: UNKNOWN to 2.6: DONTNEED, 2.7: DONTNEED, 3.0: REQUIRED

This issue has been fixed in the master branch. I bisected the fixing commit to 39a2ba5cc559900c30c3143da32446c2f20a7484. This issue exists in Ruby 3.0 but not in Ruby 2.7 or 2.6, so if possible, this should be backported to 3.0.

#11 - 07/03/2021 03:15 AM - nagachika (Tomoyuki Chikanaga)

- Backport changed from 2.6: DONTNEED, 2.7: DONTNEED, 3.0: REQUIRED to 2.6: DONTNEED, 2.7: DONTNEED, 3.0: DONE

Thank you for your investigations.
39a2ba5cc559900c30c3143da32446c2f20a7484 was already backported at d47df50678b00bd622e6be474031204ed2e52b31.
See https://bugs.ruby-lang.org/issues/17806 too.
I will fill the Backport field with "3.0: DONE".