Ruby master - Bug #14582

Unable to use `method__entry` and `method_return` tracing probes since 2.5

03/07/2018 06:23 AM - guilhermereiscampos (Guilherme Reis Campos)

Status:	Open		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	2.5	**Backport:**	2.3: UNKNOWN, 2.4: UNKNOWN, 2.5: UNKNOWN

Description

Hi,

I am trying to use dtrace/systemtap probes and not being able to use it after the 2.5. The 2.4 version works fine. I was hoping this was fixed on 2.6-preview, but apparently not (just downloaded dev and tested).

I tried on OSX using dtrace and also on ubuntu (vagrant).

```
# test.rb
class Foo

  def bar
    100.times { "Bar" }
  end
end

foo = Foo.new
foo.bar
# test.stp
probe process("/home/vagrant/.rbenv/versions/2.4.0/bin/ruby").mark("method__entry") # you will nee
d to change this to your ruby path of your version.
{
    printf("%s => %s.%s in %s:%d\n", thread_indent(1), kernel_string($arg1),kernel_string($arg2),k
ernel_string($arg3),$arg4);
}
probe process("/home/vagrant/.rbenv/versions/2.4.0/bin/ruby").mark("method__return")
{
    printf("%s <= %s.%s in %s:%d\n", thread_indent(-1), kernel_string($arg1),kernel_string($arg2),
kernel_string($arg3),$arg4);
}
```

dtrace was something similar to it.

I was expecting to see this output:

```
# lots of calls
# ....
# then:
  4090 ruby(9667):   <= Gem::Specification.unresolved_deps in /home/vagrant/.rbenv/versions/2.4.0/
lib/ruby/2.4.0/rubygems/specification.rb:1298
  4095 ruby(9667):   => MonitorMixin.mon_exit in /home/vagrant/.rbenv/versions/2.4.0/lib/ruby/2.4.
0/monitor.rb:197
  4100 ruby(9667):    => MonitorMixin.mon_check_owner in /home/vagrant/.rbenv/versions/2.4.0/lib/r
uby/2.4.0/monitor.rb:247
  4104 ruby(9667):    <= MonitorMixin.mon_check_owner in /home/vagrant/.rbenv/versions/2.4.0/lib/r
uby/2.4.0/monitor.rb:251
  4109 ruby(9667):   <= MonitorMixin.mon_exit in /home/vagrant/.rbenv/versions/2.4.0/lib/ruby/2.4.
0/monitor.rb:204
  4283 ruby(9667):  <= Kernel.require in /home/vagrant/.rbenv/versions/2.4.0/lib/ruby/2.4.0/rubyge
ms/core_ext/kernel_require.rb:55
  4303 ruby(9667): <= Kernel.require in /home/vagrant/.rbenv/versions/2.4.0/lib/ruby/2.4.0/rubygem
s/core_ext/kernel_require.rb:55
```

```
     0 ruby(9667): => Foo.bar in test.rb:3
    16 ruby(9667): <= Foo.bar in test.rb:5
```

(The output above is 2.4)

my ruby (all versions that I tested) was install with rb-env:

```
RUBY_CONFIGURE_OPTS='--enable-dtrace --disable-install-doc' rbenv install 2.5.0
```

I am happy to provide details if required. I'd also be happy to fix it if I have guidance.

Thanks,

History

#1 - 03/07/2018 06:24 AM - guilhermereiscampos (Guilherme Reis Campos)

- Description updated

#2 - 03/07/2018 06:26 AM - guilhermereiscampos (Guilherme Reis Campos)

- Description updated

#3 - 03/07/2018 06:28 AM - guilhermereiscampos (Guilherme Reis Campos)

- Description updated

#4 - 03/07/2018 11:32 AM - guilhermereiscampos (Guilherme Reis Campos)

Oh. I decided to dig a little bit deeper. Found this this: https://bugs.ruby-lang.org/issues/14104 apparently is required to do:

```
TracePoint.new{}.enable
```

I've added to my code and it works.

I was expecting that this dynamic tracing (dtrace, systemtap) would be enabled since it's compiled with --enable-dtrace and also when listing the probes all the probes are shown.

```
vagrant@vagrant-ubuntu-trusty-64:/vagrant/ruby/tracing/SystemTap$ sudo stap test.stp -L 'process("/home/vagran
t/.rbenv/versions/ruby-dev/bin/ruby").mark("*")'
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("array__create") $arg1:long $arg2:long $arg3:l
ong
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("cmethod__entry") $arg1:long $arg2:long $arg3:
long $arg4:long
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("cmethod__return") $arg1:long $arg2:long $arg3
:long $arg4:long
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("find__require__entry") $arg1:long $arg2:long
$arg3:long
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("find__require__return") $arg1:long $arg2:long
 $arg3:long
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("gc__mark__begin")
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("gc__mark__end")
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("gc__sweep__begin")
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("gc__sweep__end")
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("hash__create") $arg1:long $arg2:long $arg3:lo
ng
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("load__entry") $arg1:long $arg2:long $arg3:lon
g
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("load__return") $arg1:long $arg2:long $arg3:lo
ng
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("method__cache__clear") $arg1:long $arg2:long
$arg3:long
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("method__entry") $arg1:long $arg2:long $arg3:l
ong $arg4:long
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("method__return") $arg1:long $arg2:long $arg3:
long $arg4:long
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("object__create") $arg1:long $arg2:long $arg3:
long
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("parse__begin") $arg1:long $arg2:long
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("parse__end") $arg1:long $arg2:long
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("raise") $arg1:long $arg2:long $arg3:long
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("require__entry") $arg1:long $arg2:long $arg3:
long
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("require__return") $arg1:long $arg2:long $arg3
```

```
:long
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("string__create") $arg1:long $arg2:long $arg3:
long
process("/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby").mark("symbol__create") $arg1:long $arg2:long $arg3:
long
```

Some of the probes cases even work without the TracePoint.new{}.enable (For instance, parse_begin, parse_end)

For this script the output is:

```
TracePoint.new{}.enable
class Foo

  def bar
    100.times { "Bar" }
  end
end

foo = Foo.new
foo.bar
# sudo stap test.stp -c '/home/vagrant/.rbenv/versions/ruby-dev/bin/ruby test.rb'
# output:
#     0 ruby(27101): <= Gem::Dependency.to_specs in /home/vagrant/.rbenv/versions/ruby-dev/lib/ruby/2.6.0/ruby
gems/dependency.rb:310
#     9 ruby(27101): <= Gem::Dependency.to_spec in /home/vagrant/.rbenv/versions/ruby-dev/lib/ruby/2.6.0/rubyg
ems/dependency.rb:322
#    13 ruby(27101): <= Kernel.gem in /home/vagrant/.rbenv/versions/ruby-dev/lib/ruby/2.6.0/rubygems/core_ext/k
ernel_gem.rb:65
#  5813 ruby(27101): => Foo.bar in test.rb:5
# 5856 ruby(27101): <= Foo.bar in test.rb:6
```

#5 - 06/05/2018 03:42 AM - normalperson (Eric Wong)

guilhermekbsa@gmail.com wrote:

> Oh. I decided to dig a little bit deeper. Found this this:
> https://bugs.ruby-lang.org/issues/14104 apparently is required
> to do:
>
> ```
> TracePoint.new{}.enable
> ```
>
> I've added to my code and it works.
>
> I was expecting that this dynamic tracing (dtrace, systemtap)
> would be enabled since it's compiled with``` --enable-dtrace

Hmm... I had to make this change to get tests to pass,
but I'm not sure how I feel about it:
https://80x24.org/spew/20180605033704.GA32537@ailurophile/raw

The rest of the series to fix dtrace tests on FreeBSD 11:
https://80x24.org/spew/20180605032921.32337-1-e@80x24.org/raw
https://80x24.org/spew/20180605032921.32337-2-e@80x24.org/raw
https://80x24.org/spew/20180605032921.32337-3-e@80x24.org/raw