Ruby master - Bug #6558
Crash in garbage collection - using caller inside finalizer method
06/08/2012 03:27 AM - wpaulson (Bill Paulson)

<table>
<thead>
<tr>
<th>Status:</th>
<th>Rejected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td>authorNari</td>
</tr>
<tr>
<td>(Narihiro Nakamura)</td>
<td></td>
</tr>
<tr>
<td>Target version:</td>
<td>2.0.0</td>
</tr>
<tr>
<td>ruby -v:</td>
<td>ruby 1.9.2p290 (2011-07-09 revision 32553) [i686-linux]</td>
</tr>
</tbody>
</table>

**Description**

While debugging a hangup in a daemon, created the following program that crashes intermittently in 1.9.1 and 1.9.2 (don't have a current 1.9.3), and finishes in 1.8 (ruby 1.8.7 (2012-02-08 patchlevel 358) [i686-linux]):

```ruby
====== infinite.rb
def foo
  x = "hello world"*100000
  ObjectSpace.define_finalizer(x, lambda { |x| recurse(1000);print "x: "+caller[1]+"n"})
  x = nil
end

def recurse(n)
  y="hello world"*1000
  ObjectSpace.define_finalizer(y, lambda { |x| print "y: "+caller[1]+"n" })
  y = nil
  n>0 ? recurse(n-1): nil
end

500.times do |i|
  foo
end
puts "done"
```

The host machine is running Red Hat Enterprise Linux Server release 5.3 (Tikanga).

In 1.9, even when it finishes it's still got a problem: the number of output messages is less than the number of finalizers that should run. Running it via `ruby infinite.rb | sort | uniq -c` gets results like:

```
1 done
496 x: infinite.rb:2:in foo'
2 x: infinite.rb:8:in recurse'
105214 y: infinite.rb:2:in foo'
359123 y: infinite.rb:8:in recurse'
19099 y: infinite.rb:9:in block in recurse'
999 y: infinite.rb:9:innew'
2004 y: infinite.rb:9:in print'
12013 y: infinite.rb:9:in recurse'
```

The number of "x: infinite" messages adds to 498 (496+2), rather than the expected 500. The number of "y: infinite" messages adds up to 498452 instead of the expected 500,000.

```
========== Trace from crashing run
infinite.rb:9: [BUG] rb_gc_mark(): unknown data type 0x0(0xa3a64bc) non object
ruby 1.9.1p429 (2010-07-02 revision 28523) [i686-linux]
-- control frame --------
c:0055 p:---- s:0243 b:0243 l:000242 d:000242 CFUNC :caller
```

09/18/2021
infinite.rb:9: [BUG] object allocation during garbage collection phase
ruby 1.9.1p429 (2010-07-02 revision 28523) [i686-linux]
-- Ruby level backtrace information-------------------------------
infinite.rb:9:in caller'
infinite.rb:9:inblock in recurse'
infinite.rb:8:in call'
infinite.rb:8:in recurse'
infinite.rb:11:in recurse'
infinite.rb:11:in recurse'
infinite.rb:11:in recurse'
infinite.rb:11:in recurse'
infinite.rb:11:in recurse'
infinite.rb:11:in recurse'
infinite.rb:11:in recurse'
infinite.rb:11:in recurse'
infinite.rb:11:in recurse'
infinite.rb:11:in recurse'
 TOP

09/18/2021
You may encounter a bug of Ruby interpreter. Bug reports are welcome.
For details: [http://www.ruby-lang.org/bugreport.html](http://www.ruby-lang.org/bugreport.html)

```
1 <internal:gem_prelude>:286:in `push_all_highest_version_gems_on_load_path'
1 <internal:gem_prelude>:355:in `<compiled>'
1 <internal:gem_prelude>:38:in `dir'
1 <internal:gem_prelude>:47:in `path'
1 <internal:gem_prelude>:69:in `force_encoding'
1 <internal:gem_prelude>:69:in `set_home'
1 <internal:gem_prelude>:76:in `set_paths'
1 Error loading gem paths on load path in gem_prelude
1 can't modify frozen string
35 x: infinite.rb:2:in `foo'
 8 x: infinite.rb:8:in `recurse'
```

7002 y: infinite.rb:2:in foo'
31485 y: infinite.rb:8:in recurse'
1017 y: infinite.rb:9:in block in recurse'
2002 y: infinite.rb:9:in `print'
1 y: infinite.rb:9:in `recurse'

It does not crash for me using Ruby 1.9.3p0. I've also added a thread on top of the above script:

```ruby
Thread.new do
  loop do
    sleep 0.01
    GC.start
  end
end
```

I use it to force GC and detect C extensions errors. As said before, I cannot reproduce the crash in 1.9.3p0.

---

Can you reproduce it on the trunk?

---

nari-san, could you check it?

---

Ummm.. I can't reproduce SEGV in trunk.

Please reopen this ticket if you reproduce it.

Thanks.