Ruby trunk - Bug #10686

Memory leaking from torture test of symbol GC

01/01/2015 12:31 AM - headius (Charles Nutter)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	2.2p0	**Backport:**	2.0.0: DONTNEED, 2.1: DONTNEED, 2.2: DONE

Description

The following code appears to grow without bounds when running on MRI 2.2p0 (and grows *very* fast...hold on to your RAM):

```
x = 0; loop { (x += 1).to_s.to_sym }
```

I asked ko1 about this on Twitter and he said it appears to be leaking strings somewhere.

Associated revisions

Revision 8717a9ec - 01/01/2015 02:18 AM - normal

symbol.c: fix memory leak from global fstr hash

- symbol.c (rb_gc_free_dsymbol): delete from global fstr hash
- test/ruby/test_symbol.rb (test_symbol_fstr_leak): test for bug [ruby-core:67268] [Bug #10686]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@49090 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 49090 - 01/01/2015 02:18 AM - normalperson (Eric Wong)

symbol.c: fix memory leak from global fstr hash

- symbol.c (rb_gc_free_dsymbol): delete from global fstr hash
- test/ruby/test_symbol.rb (test_symbol_fstr_leak): test for bug [ruby-core:67268] [Bug #10686]

Revision 49090 - 01/01/2015 02:18 AM - normal

symbol.c: fix memory leak from global fstr hash

- symbol.c (rb_gc_free_dsymbol): delete from global fstr hash
- test/ruby/test_symbol.rb (test_symbol_fstr_leak): test for bug [ruby-core:67268] [Bug #10686]

Revision 49090 - 01/01/2015 02:18 AM - normal

symbol.c: fix memory leak from global fstr hash

- symbol.c (rb_gc_free_dsymbol): delete from global fstr hash
- test/ruby/test_symbol.rb (test_symbol_fstr_leak): test for bug [ruby-core:67268] [Bug #10686]

Revision 49090 - 01/01/2015 02:18 AM - normal

symbol.c: fix memory leak from global fstr hash

- symbol.c (rb_gc_free_dsymbol): delete from global fstr hash
- test/ruby/test_symbol.rb (test_symbol_fstr_leak): test for bug [ruby-core:67268] [Bug #10686]

Revision 49090 - 01/01/2015 02:18 AM - normal

symbol.c: fix memory leak from global fstr hash

- symbol.c (rb_gc_free_dsymbol): delete from global fstr hash
- test/ruby/test_symbol.rb (test_symbol_fstr_leak): test for bug [ruby-core:67268] [Bug #10686]

Revision dd52ab5a - 01/16/2015 05:58 AM - naruse (Yui NARUSE)

merge revision(s) 49090: [Backport #10686]

```
* symbol.c (rb_gc_free_dsymbol): delete from global fstr hash
```

```
* test/ruby/test_symbol.rb (test_symbol_fstr_leak): test for bug
  [ruby-core:67268] [Bug #10686]
```

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/branches/ruby_2_2@49274 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 49274 - 01/16/2015 05:58 AM - naruse (Yui NARUSE)

merge revision(s) 49090: [Backport #10686]

```
* symbol.c (rb_gc_free_dsymbol): delete from global fstr hash
```

```
* test/ruby/test_symbol.rb (test_symbol_fstr_leak): test for bug
  [ruby-core:67268] [Bug #10686]
```

Revision 251b6628 - 10/16/2015 08:35 PM - odaira

- test/ruby/test_symbol.rb (test_symbol_fstr_leak): add a warm-up code and check RSS to avoid false positive on AIX and false negative on Mac OS X. [Bug #10686]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@52140 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 52140 - 10/16/2015 08:35 PM - odaira

- test/ruby/test_symbol.rb (test_symbol_fstr_leak): add a warm-up code and check RSS to avoid false positive on AIX and false negative on Mac OS X. [Bug #10686]

Revision 52140 - 10/16/2015 08:35 PM - odaira

- test/ruby/test_symbol.rb (test_symbol_fstr_leak): add a warm-up code and check RSS to avoid false positive on AIX and false negative on Mac OS X. [Bug #10686]

Revision 52140 - 10/16/2015 08:35 PM - odaira

- test/ruby/test_symbol.rb (test_symbol_fstr_leak): add a warm-up code and check RSS to avoid false positive on AIX and false negative on Mac OS X. [Bug #10686]

Revision 52140 - 10/16/2015 08:35 PM - odaira

- test/ruby/test_symbol.rb (test_symbol_fstr_leak): add a warm-up code and check RSS to avoid false positive on AIX and false negative on Mac OS X. [Bug #10686]

Revision 52140 - 10/16/2015 08:35 PM - odaira

- test/ruby/test_symbol.rb (test_symbol_fstr_leak): add a warm-up code and check RSS to avoid false positive on AIX and false negative on Mac OS X. [Bug #10686]

Revision a6eea9be - 11/27/2015 08:49 PM - nagachika (Tomoyuki Chikanaga)

merge revision(s) 49813,52140: [Backport #10686]

```
* test/ruby/test_symbol.rb: avoid a false positive in AIX.
```

```
* test/ruby/test_symbol.rb (test_symbol_fstr_leak): add a warm-up
  code and check RSS to avoid false positive on AIX and false
  negative on Mac OS X. [Bug #10686]
```

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/branches/ruby_2_2@52770 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 52770 - 11/27/2015 08:49 PM - nagachika (Tomoyuki Chikanaga)

merge revision(s) 49813,52140: [Backport #10686]

```
* test/ruby/test_symbol.rb: avoid a false positive in AIX.
```

```
* test/ruby/test_symbol.rb (test_symbol_fstr_leak): add a warm-up
  code and check RSS to avoid false positive on AIX and false
  negative on Mac OS X. [Bug #10686]
```

History

#1 - 01/01/2015 12:35 AM - ko1 (Koichi Sasada)

Confirming code:

```
require 'objspace'
require 'pp'

def sym_num; Symbol.all_symbols.size; end
x = 0
loop {
  (x += 1).to_s.to_sym
  if (x % 1000_000) == 0
    pp ObjectSpace.count_objects
  end
}
```

We can see that T_STRING is increasing.

#2 - 01/01/2015 12:58 AM - normalperson (Eric Wong)

This seems to fix it:

```
--- a/symbol.c
+++ b/symbol.c
@@ -664,6 +664,7 @@ rb_gc_free_dsymbol(VALUE sym)
     if (str) {
     RSYMBOL(sym)->fstr = 0;
     unregister_sym(str, sym);
+    rb_hash_delete(global_symbols.dsymbol_fstr_hash, str);
     }
  }
```

http://80x24.org/spew/m/85a6346bf10d3f37fecb4926a57699fbd82b45b6.txt

#3 - 01/01/2015 01:24 AM - nobu (Nobuyoshi Nakada)

- Description updated

Rather you may want call rb_hash_delete_entry().

#4 - 01/01/2015 01:58 AM - normalperson (Eric Wong)

nobu@ruby-lang.org wrote:

> Rather you may want call rb_hash_delete_entry().

Thanks, updated

http://80x24.org/spew/m/bug10686-dsym-fstr-leak-v2@r49089.txt

#5 - 01/01/2015 02:18 AM - Anonymous

- Status changed from Open to Closed

- % Done changed from 0 to 100

Applied in changeset r49090.

---

symbol.c: fix memory leak from global fstr hash

- symbol.c (rb_gc_free_dsymbol): delete from global fstr hash
- test/ruby/test_symbol.rb (test_symbol_fstr_leak): test for bug [ruby-core:67268] [Bug #10686]

#6 - 01/01/2015 02:09 PM - nagachika (Tomoyuki Chikanaga)

- Backport changed from 2.0.0: UNKNOWN, 2.1: UNKNOWN, 2.2: UNKNOWN to 2.0.0: DONTNEED, 2.1: DONTNEED, 2.2: REQUIRED

#7 - 01/16/2015 06:02 AM - naruse (Yui NARUSE)

- Backport changed from 2.0.0: DONTNEED, 2.1: DONTNEED, 2.2: REQUIRED to 2.0.0: DONTNEED, 2.1: DONTNEED, 2.2: DONE

ruby_2_2 r49274 merged revision(s) 49090.

#8 - 01/17/2015 06:58 PM - swills (Steve Wills)

I have to disagree that this isn't needed to be back ported to 2.1. I just tried it with my ruby 2.1 and saw memory growing rather quickly. My ruby -v:

ruby 2.1.5p273 (2014-11-13 revision 48405) [amd64-freebsd11]

#9 - 01/17/2015 09:49 PM - normalperson (Eric Wong)

Symbol GC is a new feature in 2.2. In 2.1, symbols could never be GC-ed
at all so you'll see this growth, and I don't think it's the policy to
backport new features.

#10 - 01/18/2015 04:28 PM - swills (Steve Wills)

Eric Wong wrote:

> Symbol GC is a new feature in 2.2. In 2.1, symbols could never be GC-ed
> at all so you'll see this growth, and I don't think it's the policy to
> backport new features.

I see, thanks for the explanation. Not back porting makes sense then.

#11 - 09/27/2015 05:52 AM - ReiOdaira (Rei Odaira)

- File test_symbol.patch added

I would like to propose two changes in test/ruby/test_symbol.rb (TestSymbol#test_symbol_fstr_leak).

First, this test checks only the increase in the virtual size, but it should check the rss as well. On Mac OS X, the memory leak of the symbols increases the rss, not the virtual size. The following example demonstrates the increase in the rss when the fix in symbol.c is not applied. Therefore, test_symbol_fstr_leak should specify "rss: true" as an argument to assert_no_memory_leak().

```
$ ruby --disable=gems -I test/lib -r memory_status -e 'p Memory::Status.new; 200_000.times { |i| i.to_s.to_sym
 }; GC.start; p Memory::Status.new'
#<struct Memory::Status size=2505801728, rss=4321280>
#<struct Memory::Status size=2563211264, rss=47579136>
```

The second change I am proposing is to add a warm-up phase before measuring the actual memory size increase. On a certain AIX machine, test_symbol_fstr_leak causes 2.6x increase in the virtual size, even after applying the fix in symbol.c.

```
$ ruby --disable=gems -I test/lib -r memory_status -e 'p Memory::Status.new; 200_000.times { |i| i.to_s.to_sym
 }; GC.start; p Memory::Status.new'
#<struct Memory::Status size=1646592, rss=4091904>
#<struct Memory::Status size=4263936, rss=6725632>
```

However, I don't think this is a leak, because if you sequentially run a similar test twice and only measure the second run, neither the virtual size or the rss increases so much, as follows.

```
$ ruby --disable=gems -I test/lib -r memory_status -e '200_000.times { |i| i.to_s.to_sym }; GC.start; p Memory
::Status.new; 200_000.times { |i| (i+200_000).to_s.to_sym }; GC.start; p Memory::Status.new'
#<struct Memory::Status size=4276224, rss=6758400>
#<struct Memory::Status size=4321280, rss=6803456>
```

Because the warm-up phase inevitably increases memory usage, we should measure the steady state after the warm-up phase when detecting a memory leak.

The attached patch includes these two changes. I confirmed my patch on Mac OS X, POWER Linux, and AIX. Without the fix in symbol.c, test_symbol_fstr_leak correctly fails on every environment, and with the fix in symbol.c, it succeeds.

#12 - 10/27/2015 07:30 AM - nagachika (Tomoyuki Chikanaga)

Applied at r52140.

#13 - 10/27/2015 07:33 AM - nagachika (Tomoyuki Chikanaga)

- Backport changed from 2.0.0: DONTNEED, 2.1: DONTNEED, 2.2: DONE to 2.0.0: DONTNEED, 2.1: DONTNEED, 2.2: REQUIRED

#14 - 11/27/2015 08:49 PM - nagachika (Tomoyuki Chikanaga)

- Backport changed from 2.0.0: DONTNEED, 2.1: DONTNEED, 2.2: REQUIRED to 2.0.0: DONTNEED, 2.1: DONTNEED, 2.2: DONE

Backported r49813 and r52140 into ruby_2_2 branch at r52770.

Files

test_symbol.patch	610 Bytes	09/27/2015	ReiOdaira (Rei Odaira)