Ruby 1.8 - Bug #3416

[patch] ruby_1_8_7 [BUG] gc_sweep(): unknown data type 0x0

06/09/2010 01:22 PM - mattgleeson (Matt Gleeson)

Status:	Open
Priority:	Normal
Assignee:	
Target version:	
ruby -v:	

Description

=begin
I've been having a problem with my rails app that did not appear in 1.8.7-p174 but does appear in 1.8.7-p248 and current ruby_1_8_7 from svn. Unfortunately I have not been able to make a nice small test case to reproduce it. It appears to have something to with taint and marshal interaction. I use tainting a lot in my security tests and it goes away if I turn off the extra tainting.

I see this message:

trunk/vendor/rails/actionpack/lib/action_view/helpers/debug_helper.rb:33: [BUG] gc_sweep(): unknown data type 0x0(0xb61cad34)
ruby 1.8.7 (2010-06-07 patchlevel 268) [i686-linux]

Aborted

My stack trace at that point looks like this:

(gdb) where
#0 0x0012d422 in **kernel_vsyscall ()**
#1 0x001bd651 in *GI_raise (sig=6) at ../nptl/sysdeps/unix/sysv/linux/raise.c:64
#2 0x001c0a82 in *__GI_abort () at abort.c:92
#3 0x080e5cae in rb_bug (fmt=0x80e8c24 "gc_sweep(): unknown data type 0x%lx(0x%lx)") at error.c:213
#4 0x08074b30 in obj_free () at gc.c:1379
#5 gc_sweep () at gc.c:1175
#6 garbage_collect () at gc.c:1524
#7 0x08074eb5 in ruby_xmalloc (size=129) at gc.c:147
#8 0x080beac6 in rb_str_buf_new (capa=128) at string.c:234
#9 0x080c0294 in rb_str_buf_new2 (ptr=0x80e914e "{") at string.c:247
#10 0x0807902b in inspect_hash (hash=3065366280) at hash.c:1319
#11 0x080d1038 in inspect_call (arg=0xbffa0674) at array.c:1487
#12 0x08056a99 in rb_ensure (b_proc=0x80d1020 , data1=3220833908, e_proc=0x80d68b0 , data2=3065366280) at eval.c:5571
#13 0x080d4fd2 in rb_protect_inspect (func=0x8079010 , obj=3065366280, arg=0) at array.c:1544
#14 0x08077a54 in rb_hash_inspect (hash=3065366280) at hash.c:1341
#15 0x08061d85 in rb_call0 (klass=, recv=, id=, oid=3145, argc=0, argv=0x0, body=0xb7fe1ed0, flags=0) at eval.c:5928
#16 0x08061f19 in rb_call (klass=3086884940, recv=3065366280, mid=3145, argc=0, argv=0x0, scope=1, self=6) at eval.c:6176
#17 0x080629f4 in vafuncall (recv=, mid=, n=, ar=0xbffa091c) at eval.c:6253
#18 0x08062b10 in rb_funcall (recv=3065366280, mid=3145, n=0) at eval.c:6270
#19 0x0808bc0a in rb_inspect (obj=3065366280) at object.c:334
#20 0x0808bdf1 in inspect_i (id=65570, value=3065366280, str=3064389560) at object.c:359
#21 0x080755e7 in foreach_safe_i (key=6, value=3065366280, arg=0xd15) at hash.c:145
#22 0x080bcc43 in st_foreach (table=0xb1411e0, func=0x80755c0 , arg=3220834804) at st.c:487
#23 0x080794ef in st_foreach_safe (table=0xb1411e0, func=0x808bd30 , a=3064389560) at hash.c:163
#24 0x0808b4e9 in inspect_obj (obj=3065430360, str=3064389560) at object.c:370
#25 0x080d1038 in inspect_call (arg=0xbffa0b44) at array.c:1487
#26 0x08056a99 in rb_ensure (b_proc=0x80d1020 , data1=3220835140, e_proc=0x80d68b0 , data2=3065430360) at eval.c:5571
#27 0x080d4fd2 in rb_protect_inspect (func=0x808b4c0 , obj=3065430360, arg=3064389560) at array.c:1544
#28 0x0808daac in rb_obj_inspect (obj=3065430360) at object.c:413
...

I don't fully understand what is happening but it looks to me like the garbage collector eventually tries to handle an object that has only the TAINT flag set (flags = 256) without an object type. This seems to make the GC think the object needs to be freed but then it can't figure out how without the type.

This change seems to avoid the problem for me:

diff --git a/marshal.c b/marshal.c
index 0112257..859bcd9 100644
--- a/marshal.c
+++ b/marshal.c
@@ -715,7 +715,10 @@ clear_dump_arg(arg)
st_free_table(arg->symbols);
arg->symbols = 0;
st_free_table(arg->data);

- if (arg->taint) {
- /* Somehow we can sometimes get here and taint a "string" that has
- already been freed, which causes it to eventually die in
- gc_sweep because flags != 0 but it doesn't have a TYPE. */
- if (arg->taint && TYPE(arg->str)) { OBJ_TAINT(arg->str); } }

I realize this change is probably not "correct" but I am new to ruby and would appreciate any advice from the experts.
=end

History

#1 - 06/16/2010 01:14 AM - zdennis (Zach Dennis)

=begin
I see this problem a lot as well with 1.8.7 p249.
=end

#2 - 07/13/2010 06:02 PM - zimbatm (zimba tm)

=begin
There is a related ticket in the 1.9 branch which seems to have been fixed. See : http://redmine.ruby-lang.org/issues/show/3463

=end

#3 - 01/06/2011 08:52 AM - tmm1 (Aman Gupta)

=begin
This bug was introduced in p226 with http://redmine.ruby-lang.org/repositories/revision/5?rev=26076, and fixed in p298 with http://redmine.ruby-lang.org/repositories/revision/5?rev=28406

Since it is no longer an issue with the last two release of 1.8.7, this bug can be closed.
=end