Ruby - Bug #15821

ruby_process_options() may cause "WB miss (O->Y)"

05/04/2019 05:42 AM - wanabe (_ wanabe)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	ruby 2.7.0dev (2019-05-04 trunk b72623012d) [x86_64-linux]	**Backport:**	2.4: REQUIRED, 2.5: DONE, 2.6: DONE

Description

Problem

Ruby interpreter may cause error "WB miss (O->Y)" on some conditions that are RGENGC_CHECK_MODE=5 and RUBY_DEBUG=gc_stress

How to reproduce

1. build ruby with high RGENGC_CHECK_MODE
   - make ruby optflags="-O3 -DRGENGC_CHECK_MODE=5"
2. run ruby with gc_stress
   - RUBY_DEBUG=gc_stress ./ruby --disable-gems -ve 1

Probable cause

1. rb_construct_expanded_load_path calls rb_ary_replace(vm->load_path_snapshot, vm->load_path).
2. It creates shared root array and makes vm->load_path SHARED_ARRAY.
3. After a while, process_options calls RARRAY_ASET(load_path, i, path).
4. It calls rb_gc_writebarrier -> gc_writebarrier_generational.
   - Incremental mark phase is finished because of RUBY_DEBUG=gc_stress.
5. It makes vm->load_path remembered, but not shared root array!
6. "WB miss (O->Y)" is done.
   - Old parent is shared root array.
   - New child is path of above 3.

Proposal

How about call rb_ary_modify before RARRAY_SET in process_options?
Or using rb_ary_store instead of RARRAY_SET may avoid the error.

Sample output

An example of full output is attached.
(Sorry, I GZipped it because of file-size limitation)
The snippet is here:

```
ruby 2.7.0dev (2019-05-04 trunk b72623012d) [x86_64-linux]
verify_internal_consistency_reachable_i: WB miss (O->Y) 0x000055c3262f3610 [3LM   ] T_ARRAY [   ]
len: 20, capa:2 ptr:0x000055c326498380 -> 0x000055c3262f3908 [2  P  ] T_STRING (String) /home/wana
be/.rbenv/versions/trunk/lib/ruby/site_ruby/2.7.0
[all refs] (size: 5307)
(snip)
[allrefs_dump_i] 0x000055c3263349f8 [3LMP  ] T_ARRAY [E ] len: 0 (embed) <- <0x000055c326336f28 [0
  P U] VM/thread (Thread) VM/thread>
./ruby: [BUG] Segmentation fault at 0x0000000000000010
ruby 2.7.0dev (2019-05-04 trunk b72623012d) [x86_64-linux]

-- Control frame information -----------------------------------------------
c:0001 p:0000 s:0003 E:0022c0 (none) [FINISH]
```

```
-- Machine register context ------------------------------------------------
 RIP: 0x000055c32452e15a RBP: 0x0000000000000001 RSP: 0x00007ffea126d470
 RAX: 0x0000000000000000 RBX: 0x000055c3262ef3c8 RCX: 0x0000000000000001
 RDX: 0x000055c324773446 RDI: 0x00007ff8c77cb680 RSI: 0x0000000000000001
  R8: 0x000055c3262ef3b8  R9: 0x0000000000000018 R10: 0x0000000000000018
 R11: 0x0000000000000246 R12: 0x0000000000000100 R13: 0x0000000000000005
 R14: 0x000055c3262f3c28 R15: 0x000055c3262ef1b0 EFL: 0x0000000000010206

-- C level backtrace information -------------------------------------------
/home/wanabe/work/prog/ruby/ruby/tmp/trunk/ruby(rb_vm_bugreport+0x554) [0x55c324769fa4] ../../vm_d
ump.c:715
[0x55c324760088]
/home/wanabe/work/prog/ruby/ruby/tmp/trunk/ruby(sigsegv+0x42) [0x55c324640d42] ../../signal.c:997
/lib/x86_64-linux-gnu/libpthread.so.0(__restore_rt+0x0) [0x7ff8c797ff40]
/home/wanabe/work/prog/ruby/ruby/tmp/trunk/ruby(allrefs_dump+0x1a) [0x55c32452e15a] /usr/include/x
86_64-linux-gnu/bits/stdio2.h:100
[0x55c32453a478]
[0x55c32453a64c]
[0x55c32453f874]
/home/wanabe/work/prog/ruby/ruby/tmp/trunk/ruby(rb_str_dup+0x29) [0x55c32465aa59] ../../string.c:7
22
[0x55c32463f2e1]
/home/wanabe/work/prog/ruby/ruby/tmp/trunk/ruby(ruby_process_options+0xc0) [0x55c3246404a0] ../../
ruby.c:2380
/home/wanabe/work/prog/ruby/ruby/tmp/trunk/ruby(ruby_options+0xca) [0x55c32451e1ea] ../../eval.c:1
18
/home/wanabe/work/prog/ruby/ruby/tmp/trunk/ruby(main+0x67) [0x55c324519ec7] ../../main.c:42
(snip)
Aborted (core dumped)
```

Associated revisions

Revision b165bedcbd41d791a85fc1ce90b57a0d0525f319 - 05/18/2019 03:17 AM - ko1 (Koichi Sasada)

skip a test to pass CIs.

I'm debugging [Bug #15821] but my patch introduces another issue.
So I simply skip this test and re-enable it later.

Revision b165bedcbd41d791a85fc1ce90b57a0d0525f319 - 05/18/2019 03:17 AM - ko1 (Koichi Sasada)

skip a test to pass CIs.

I'm debugging [Bug #15821] but my patch introduces another issue.
So I simply skip this test and re-enable it later.

Revision b165bedc - 05/18/2019 03:17 AM - ko1 (Koichi Sasada)

skip a test to pass CIs.

I'm debugging [Bug #15821] but my patch introduces another issue.
So I simply skip this test and re-enable it later.

Revision ac00bdc8a8ac2c62a94dd36a7784d15bbcb7df19 - 05/21/2019 05:17 AM - nobu (Nobuyoshi Nakada)

Do not modify shared array

[Bug #15821]

Revision ac00bdc8a8ac2c62a94dd36a7784d15bbcb7df19 - 05/21/2019 05:17 AM - nobu (Nobuyoshi Nakada)

Do not modify shared array

[Bug #15821]

Revision ac00bdc8 - 05/21/2019 05:17 AM - nobu (Nobuyoshi Nakada)

Do not modify shared array

[Bug #15821]

Revision 09b8eddcf238d795b96f15d21adcc6d9d51f7d71 - 07/31/2019 02:32 PM - nagachika (Tomoyuki Chikanaga)

merge revision(s) b165bedcbd41d791a85fc1ce90b57a0d0525f319,ac00bdc8a8ac2c62a94dd36a7784d15bbcb7df19: [Backport #15821]

```
    skip a test to pass CIs.

    I'm debugging [Bug #15821] but my patch introduces another issue.
    So I simply skip this test and re-enable it later.

    Do not modify shared array

    [Bug #15821]
```

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/branches/ruby_2_6@67720 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 09b8eddcf238d795b96f15d21adcc6d9d51f7d71 - 07/31/2019 02:32 PM - nagachika (Tomoyuki Chikanaga)

merge revision(s) b165bedcbd41d791a85fc1ce90b57a0d0525f319,ac00bdc8a8ac2c62a94dd36a7784d15bbcb7df19: [Backport #15821]

```
    skip a test to pass CIs.

    I'm debugging [Bug #15821] but my patch introduces another issue.
    So I simply skip this test and re-enable it later.

    Do not modify shared array

    [Bug #15821]
```

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/branches/ruby_2_6@67720 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 09b8eddc - 07/31/2019 02:32 PM - nagachika (Tomoyuki Chikanaga)

merge revision(s) b165bedcbd41d791a85fc1ce90b57a0d0525f319,ac00bdc8a8ac2c62a94dd36a7784d15bbcb7df19: [Backport #15821]

```
    skip a test to pass CIs.

    I'm debugging [Bug #15821] but my patch introduces another issue.
    So I simply skip this test and re-enable it later.

    Do not modify shared array

    [Bug #15821]
```

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/branches/ruby_2_6@67720 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 2576b8b26a34615e23ee43eef5b280b57e162c59 - 08/26/2019 03:02 PM - U.Nakamura

merge revision(s) ac00bdc8a8ac2c62a94dd36a7784d15bbcb7df19: [Backport #15821]

```
    Do not modify shared array

    [Bug #15821]
```

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/branches/ruby_2_5@67759 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 2576b8b26a34615e23ee43eef5b280b57e162c59 - 08/26/2019 03:02 PM - U.Nakamura

merge revision(s) ac00bdc8a8ac2c62a94dd36a7784d15bbcb7df19: [Backport #15821]

```
    Do not modify shared array

    [Bug #15821]
```

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/branches/ruby_2_5@67759 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 2576b8b2 - 08/26/2019 03:02 PM - U.Nakamura

merge revision(s) ac00bdc8a8ac2c62a94dd36a7784d15bbcb7df19: [Backport #15821]

```
    Do not modify shared array

    [Bug #15821]
```

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/branches/ruby_2_5@67759 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 05/05/2019 04:25 AM - ko1 (Koichi Sasada)

I didn't understand completely, but it is local problem on it? or all of rb_ary_replace() calls have the same issue?

#2 - 05/05/2019 09:58 AM - methodmissing (Lourens Naudé)

As outlined in the proposal, the following diff fixes the WB miss local to process_options

```
diff --git a/ruby.c b/ruby.c
index bf94b8ee13..b7648934d6 100644
--- a/ruby.c
+++ b/ruby.c
@@ -1726,6 +1726,7 @@ process_options(int argc, char **argv, ruby_cmdline_options_t *opt)
             path = rb_enc_associate(rb_str_dup(path), lenc);
 #endif
             if (mark) rb_ivar_set(path, id_initial_load_path_mark, path);
+            rb_ary_modify(load_path);
             RARRAY_ASET(load_path, i, path);
         }
     }
```

```
lourens@CarbonX1:~/src/ruby/ruby$ RUBY_DEBUG=gc_stress ./ruby --disable-gems -ve 1
ruby 2.7.0dev (2019-05-05 trunk 594a033ff0) [x86_64-linux]
gc_check_after_marks_i: 0x000055a590d68018 [3  P  ] T_STRING (String) pp is not marked and not oldgen.
gc_check_after_marks_i: 0x55a590d68018 is referred from <root@machine_context> (marked from machine stack).
-e:1: warning: possibly useless use of a literal in void context
```

Running make check with GC stress enabled comes back mostly clean (ran it for a few minutes) except for these warnings, which as I understand can be false positives:

```
./revision.h unchanged
generating encdb.h
gc_check_after_marks_i: 0x000055607246c008 [0  P  ] T_STRING format is not marked and not oldgen.
gc_check_after_marks_i: 0x55607246c008 is referred from <root@machine_context> (marked from machine stack).
gc_check_after_marks_i: 0x00005560723c4010 [3  P  ] T_STRING (String)  is not marked and not oldgen.
gc_check_after_marks_i: 0x5560723c4010 is referred from <root@machine_context> (marked from machine stack).
gc_check_after_marks_i: 0x00005560723c4010 [3  P  ] T_STRING (String)
 is not marked and not oldgen.
gc_check_after_marks_i: 0x5560723c4010 is referred from <root@machine_context> (marked from machine stack).
gc_check_after_marks_i: 0x00005560723c4010 [3  P  ] T_STRING (String) CGI is not marked and not oldgen.
gc_check_after_marks_i: 0x5560723c4010 is referred from <root@machine_context> (marked from machine stack).
gc_check_after_marks_i: 0x00005560723c4010 [2  P  ] T_STRING (String) time is not marked and not oldgen.
gc_check_after_marks_i: 0x5560723c4010 is referred from <root@machine_context> (marked from machine stack).
gc_check_after_marks_i: 0x00005560723c4010 [3  P  ] T_STRING (String)     t = time.clone.gmtime
 is not marked and not oldgen.
gc_check_after_marks_i: 0x5560723c4010 is referred from <root@machine_context> (marked from machine stack).
gc_check_after_marks_i: 0x00005560723c4010 [3  P  ] T_STRING (String)   end
 is not marked and not oldgen.
gc_check_after_marks_i: 0x5560723c4010 is referred from <root@machine_context> (marked from machine stack).
gc_check_after_marks_i: 0x00005560723c4010 [3  P  ] T_STRING (String)   # +string+ is the HTML string to inden
t.  +shift+ is the indentation
```

#3 - 05/05/2019 10:38 AM - wanabe (_ wanabe)

ko1 (Koichi Sasada) wrote:

> it is local problem on it?

I guess, no. It is not a local problem.
It will cause by RARRAY_ASET with shared array.

> or all of rb_ary_replace() calls have the same issue?

Not only rb_ary_replace() has the issue but also the array.c functions that call ary_make_shared() do.
For example, rb_ary_aref() == Array#[] can calls ary_make_shared() when Range or 2 args are passed.

But I guess "RARRAY_ASET with shared array" pattern is rare case.
Most cases in array.c use ARY_SET macro that has assert(!ARY_SHARED_P(a)) guard.
Other cases in ruby repository are not able to be "shared array" AFAIK.
For example, enum.c has RARRAY_ASET(data->buf, data->n*2, v) but data->buf can't be shared array.

C extensions may have the issue.

For example:

```
static VAUE foo(VALUE self, VALUE ary) {
  RARRAY_ASET(ary, 0, rb_str_new("foo", 3));
  return Qnil;
}

void Init_foo {
  rb_define_method(cFoo, "foo", foo, 1);
}
```

Foo#foo may cause "WB miss (O->Y)" when old shared array is given.

#4 - 05/21/2019 01:56 PM - nobu (Nobuyoshi Nakada)

- Backport changed from 2.4: UNKNOWN, 2.5: UNKNOWN, 2.6: UNKNOWN to 2.4: REQUIRED, 2.5: REQUIRED, 2.6: REQUIRED

- Status changed from Open to Closed

Closed by ac00bdc8a8ac2c62a94dd36a7784d15bbcb7df19

#5 - 05/22/2019 08:35 AM - ko1 (Koichi Sasada)

Note of this issue:

We assume that ary in RARRAY_ASET(ary, ...) is not FROZEN and SHARED. This code violate this assumption.
Nobu's fix solved this issue by using rb_ary_push() which works with a SHARED array (and added a trick for LOAD_PATH feature).
rb_ary_store() is also acceptable, in this case.

In general, we assume C-extension writers don't violate this assumption. But RARRAY_ASET() is easy to misusing.
There are several options to take:

- (1) Do nothing (now!)
- (2) call rb_ary_modify() in RARRAY_ASET()
- (3) replace with rb_ary_store() by macro (#define RARRAY_ASET rb_ary_store)
- (4) check SHARED and FROZEN, then call rb_bug()

(2) introduces additional cost, but make it safer.
(3) introduces more aggressive check (range checks).
(4) can stop application with error because the usage of RARRAY_ASET is wrong.

I think (2) is acceptable for general cases of RARRAY_ASET, now.

#6 - 07/31/2019 02:32 PM - nagachika (Tomoyuki Chikanaga)

- Backport changed from 2.4: REQUIRED, 2.5: REQUIRED, 2.6: REQUIRED to 2.4: REQUIRED, 2.5: REQUIRED, 2.6: DONE

ruby_2_6 r67720 merged revision(s) b165bedcbd41d791a85fc1ce90b57a0d0525f319,ac00bdc8a8ac2c62a94dd36a7784d15bbcb7df19.

#7 - 08/26/2019 03:02 PM - usa (Usaku NAKAMURA)

- Backport changed from 2.4: REQUIRED, 2.5: REQUIRED, 2.6: DONE to 2.4: REQUIRED, 2.5: DONE, 2.6: DONE

ruby_2_5 r67759 merged revision(s) ac00bdc8a8ac2c62a94dd36a7784d15bbcb7df19.

Files

out.log.gz	114 KB	05/04/2019	wanabe (_ wanabe)