Ruby master - Bug #18133
LTO: TestGCCompact#test_ast_compacts segfaults on i686
08/25/2021 03:02 PM - vo.x (Vit Ondruch)

Status: Assigned
Priority: Normal
Assignee:
Target version:
ruby -v: ruby 3.0.2p107 (2021-07-07 revision 0db68f0223) [i386-linux]
Backport: 2.6: UNKNOWN, 2.7: UNKNOWN, 3.0: UNKNOWN

Description
I observe following segfault running the test suite on i686 on RHEL9:

```
$ gdb --args ./miniruby -I./lib -I.-I.ext/common ./tool/runruby.rb --extout=.ext -- --disable-gems ./test/runner.rb --excludes-dir=./test/excludes -v

... snip ...
(gdb) handle SIGPIPE noprint nostop pass
Signal Stop Print Pass to program Description
SIGPIPE No No Yes Broken pipe (gdb) r

... snip ...
[ 8347/20497] TestGBK#test_mbc_enc_len = 0.00 s
[ 8348/20497] TestGBK#test_mbc_to_code = 0.00 s
[ 8349/20497] TestGCCompact#test_ast_compacts--Type <RET> for more, q to quit, c to continue without paging--
Thread 1 "ruby" received signal SIGSEGV, Segmentation fault.
0xf7e33fe6 in rb_class_remove_from_super_subclasses (klass=<optimized out>) at /build-dir/build/BUILD/ruby-3.0.2/class.c:96
96     RCLASS_EXT(entry->next->klass)->parent_subclasses = RCLASS_EXT(klass)->parent_subclasses;
(gdb) bt
#0 0xf7e33fe6 in rb_class_remove_from_super_subclasses (klass=<optimized out>) at /build-dir/build/BUILD/ruby-3.0.2/class.c:96
#1 obj_free (obj=<optimized out>, objspace=0x5655ac30) at /build-dir/build/BUILD/ruby-3.0.2/gc.c:3019
#2 gc_page_sweep (sweep_page=0x5a40e1f0, heap=0x5655ac48, objspace=0x5655ac30) at /build-dir/build/BUILD/ruby-3.0.2/gc.c:4914
#3 gc_sweep_step.isra.0 (objspace=<optimized out>, heap=<optimized out>) at /build-dir/build/BUILD/ruby-3.0.2/gc.c:5134
#4 0xf7ca3f09 in gc_sweep_rest (objspace=<optimized out>) at /build-dir/build/BUILD/ruby-3.0.2/gc.c:5190
#5 gc_sweep (objsweep=0x5655ac30) at /build-dir/build/BUILD/ruby-3.0.2/gc.c:5313
#6 0xf7ca8250 in gc_marks (full_mark=<optimized out>, objspace=<optimized out>) at /build-dir/build/BUILD/ruby-3.0.2/gc.c:7504
#7 gc_start (objspace=<optimized out>, reason=<optimized out>) at /build-dir/build/BUILD/ruby-3.0.2/gc.c:8210
#8 0xf7ca8530 in garbage_collect (objspace=objspace@entry=0x5655ac30, reason=reason@entry=238592) at /build-dir/build/BUILD/ruby-3.0.2/gc.c:8210
#9 0xf7ca723 in gc_start_internal (compact=2, immediate_sweep=2, immediate_mark=2, full_mark=2, self=1448715280, ec=0x5655afac) at /build-dir/build/BUILD/ruby-3.0.2/gc.c:8553
#10 gc_compact (ec=0x5655afac, self=1448715280) at /build-dir/build/BUILD/ruby-3.0.2/gc.c:9468
#11 0xf7dfae3c in invoke_bf (argv=0x0, bf=<optimized out>, reg_cfp=<optimized out>, ec=0x5655afac) at /build-dir/build/BUILD/ruby-3.0.2/vm_insnhelper.c:5583
#12 vm_invoke_builtin_delegate (ec=0x5655afac, cf=<optimized out>, bf=<optimized out>, start_index=x) at /build/dir/build/BUILD/ruby-3.0.2/vm_insnhelper.c:5583
#13 0xf7e0664c in vm_exec_core (ec=0x0, initial=1448732852) at /build/dir/build/BUILD/ruby-3.0.2/vm_insnhelper.c:1482
#14 0xf7e1d0d5 in rb_vm_exec (ec=<optimized out>, mjit_enable_p=<optimized out>) at /build/dir/build/Build
```

09/01/2022
Unfortunately:

1. I don’t have better reproducer than to run the whole test suite and even then it is not triggered always. I was not successful to hit the issue running just the single test case or the test file.
2. I have failed to reproduce this on CentOS Stream 9, which is surprising.

Luckily, I can reproduce it on my system.

This is seems to be related to LTO, because I have never faced such issue with LTO disabled.

Related issues:

| #1 | 08/25/2021 03:08 PM - peterzhu2118 (Peter Zhu) | The backtrace looks similar to #18119 which is triggered in Ractor. |
| #2 | 08/25/2021 03:30 PM - vo.x (Vit Ondruch) | It actually crashes on c9s: |
#3 - 11/26/2021 08:24 AM - vo.x (Vit Ondruch)
Not sure if I was previously lucky on Fedora, but trying to update to Ruby 3.0.3, from 10 builds I have made 8 failed due to this issue.

#4 - 12/07/2021 07:15 PM - peterzhu2118 (Peter Zhu)
- Status changed from Open to Closed
Hi, I was able to debug a core dump for this bug. Backports in #18394 should fix it. Thanks for the bug report!

#5 - 12/08/2021 12:47 PM - vo.x (Vit Ondruch)
- Status changed from Closed to Assigned
Thanks for looking into this. However, applying these two patches, while fixing i686, it breaks ppc64le :(

```
[ 8890/21266] TestGCCompact=test_ast_compacts<internal:gc>:213: [BUG] Couldn't unprotect page 0x000000140f9800
ruby 3.0.3p157 (2021-11-24 revision 3fb7d2cadc) [powerpc64le-linux]
-- Control frame information -----------------------------------------------
c:0031 p:0003 s:0175 e:000174 METHOD <internal:gc>:213
  c:0020 p:0006 s:0094 e:000104 CFUNC :map
  c:0017 p:0008 s:0047 e:000104 CFUNC :each
  c:0006 p:0006 s:0000 e:000098 METHOD /builddir/build/BUILD/ruby-3.0.3/tool/lib/minitest/unit.rb:1126
  c:0005 p:0006 s:0000 e:000098 METHOD /builddir/build/BUILD/ruby-3.0.3/tool/lib/minitest/unit.rb:1125
  c:0004 p:0006 s:0000 e:000098 METHOD /builddir/build/BUILD/ruby-3.0.3/tool/lib/minitest/unit.rb:1124
  c:0002 p:0006 s:0000 e:000098 METHOD /builddir/build/BUILD/ruby-3.0.3/tool/lib/minitest/unit.rb:1122
  c:0001 p:0006 s:0000 e:000098 METHOD /builddir/build/BUILD/ruby-3.0.3/tool/lib/minitest/unit.rb:1121
-- Ruby level backtrace information ----------------------------------------
./test/runner.rb:11:in `<main>'
./test/runner.rb:11:in `require_relative'
/builddir/build/BUILD/ruby-3.0.3/tool/test/runner.rb:23:in `<top (required)>'
/builddir/build/BUILD/ruby-3.0.3/tool/lib/test/unit.rb:1249:in `run'
/builddir/build/BUILD/ruby-3.0.3/tool/lib/test/unit.rb:1245:in `run'
/builddir/build/BUILD/ruby-3.0.3/tool/lib/test/unit.rb:1175:in `run'
/builddir/build/BUILD/ruby-3.0.3/tool/lib/test/unit.rb:34:in `run'
/builddir/build/BUILD/ruby-3.0.3/tool/lib/test/unit.rb:695:in `run'
/builddir/build/BUILD/ruby-3.0.3/tool/lib/test/unit.rb:847:in `run'
```
09/01/2022
However, it is probably fixed in master, because I have not hit this issue while testing Ruby 3.1.0.

For the time being, the build is available [here](...) and this is the [build log](...).

#6 - 12/08/2021 01:04 PM - vo.x (Vit Ondruch)

vo.x (Vit Ondruch) wrote in [note-5]:

> Thanks for looking into this. However, applying these two patches, while fixing i686, it breaks ppc64le :(

And sometimes aarch64

#7 - 12/08/2021 02:16 PM - peterzhu2118 (Peter Zhu)

Hey @vo.x (Vit Ondruch), can you check if also backporting this PR fixes the crashes? [https://github.com/ruby/ruby/pull/4227](...)

Ruby 3.0 is still using posix_memalign to allocate pages. Only memory allocated with mmap is allowed to be passed into mprotect.

#8 - 12/08/2021 04:52 PM - vo.x (Vit Ondruch)

Unfortunately, the build fails already in miniruby:

... snip ...

This is the build and full build log.

#9 - 12/08/2021 05:03 PM - peterzhu2118 (Peter Zhu)

I don't have access to a ppc64 machine. Do you know what the crash is?

#10 - 12/09/2021 11:19 AM - vo.x (Vit Ondruch)
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./miniruby...
warning: File "/builddir/build/BUILD/ruby-3.0.3/.gdbinit" auto-loading has been declined by your 'auto-load safe-path' set to "$debugdir:$datadir/auto-load".
To enable execution of this file add
add-auto-load-safe-path /builddir/build/BUILD/ruby-3.0.3/.gdbinit
line to your configuration file "/builddir/.config/gdb/gdbinit".
To completely disable this security protection add
set auto-load safe-path /
line to your configuration file "/builddir/.config/gdb/gdbinit".
For more information about this security protection see the
"Auto-loading safe path" section in the GDB manual. E.g., run from the shell:
info "/(gdb) Auto-loading safe path"
(gdb) r
Starting program: /builddir/build/BUILD/ruby-3.0.3/miniruby -I./lib -I. -I.ext/common ./tool/generic_erb.rb -o builtin_binary.inc ./builtin_binary.inc.tmpl 
To enable execution of this file add
add-auto-load-safe-path /builddir/build/BUILD/ruby-3.0.3/.gdbinit
line to your configuration file "/builddir/.config/gdb/gdbinit".
To completely disable this security protection add
set auto-load safe-path /
line to your configuration file "/builddir/.config/gdb/gdbinit".
For more information about this security protection see the
"Auto-loading safe path" section in the GDB manual. E.g., run from the shell:
info "/(gdb) Auto-loading safe path"
(gdb) bt
#0 heap_page_allocate (objectspace=0x1004b1400) at gc.c:1870
#1 heap_page_create (objectspace=0x1004b1400) at gc.c:1910
#2 heap_assign_page (objectspace=0x1004b1400, heap=0x1004b1428) at gc.c:1935
#3 0x000000001000df220 in heap_add_pages (add=24, heap=0x1004b1428, objectspace=0x1004b1400) at gc.c:1948
#4 Init_heap () at gc.c:3173
#5 ruby_setup () at eval.c:87
#6 0x000000001000e50e8 in ruby_init () at eval.c:110
#7 0x0000000010003f2fa0 in main (argc=<optimized out>, argv=<optimized out>) at ./main.c:49
(gdb) 1 heap_page_allocate
1801  struct heap_page_body *page_body = 0;
1802  size_t hi, lo, mid;
1803  int limit = HEAP_PAGE_OBJ_LIMIT;
1804  /* assign heap_page body (contains heap_page_header and RVALUEs) */
1805  page_body = (struct heap_page_body *)rb_aligned_malloc(HEAP_PAGE_ALIGN, HEAP_PAGE_SIZE);
1806  if (page_body == 0) {
1807    rb_memerror();
1808  }
1809 }
1810 (gdb)
1811 /* assign heap_page entry */
1812 page = calloc1(sizeof(struct heap_page));
1813 if (page == 0) {
1814  rb_aligned_free(page, HEAP_PAGE_SIZE);
1815  rb_memerror();
1816 }
1817 /* adjust obj_limit (object number available in this page) */
1818 start = (VALUE*)((VALUE)page + sizeof(struct heap_page_header));
if ((VALUE)start % sizeof(RVALUE) != 0) {

   int delta = (int)(sizeof(RVALUE) - ((VALUE)start % sizeof(RVALUE)));
   start = (RVALUE*)((VALUE)start + delta);
   limit = (HEAP_PAGE_SIZE - (int)((VALUE)start - (VALUE)page_body))/(int)sizeof(RVALUE);
}

   end = start + limit;

   /* setup heap_pages_sorted */
   lo = 0;
   hi = heap_allocated_pages;
   while (lo < hi) {

      struct heap_page *mid_page;

      mid = (lo + hi) / 2;
      mid_page = heap_pages_sorted[mid];
      if (mid_page->start < start) {
         lo = mid + 1;
      } else if (mid_page->start > start) {
         hi = mid;
      } else {
         rb_bug("same heap page is allocated: %p at %"PRIuVALUE, (void *)page_body, (VALUE)mid);
      }
   }

   if (hi < heap_allocated_pages) {
      MEMMOVE(&heap_pages_sorted[hi+1], &heap_pages_sorted[hi], struct heap_page_header*, heap_allocated_pages - hi);
   }

   heap_pages_sorted[hi] = page;

   heap_allocated_pages++;

   GC_ASSERT(heap_eden->total_pages + heap_allocatable_pages <= heap_pages_sorted_length);
   GC_ASSERT(heap_eden->total_pages + heap_tomb->total_pages == heap_allocated_pages - 1);
   GC_ASSERT(heap_allocated_pages <= heap_pages_sorted_length);

   objspace->profile.total_allocated_pages++;

   if (heap_allocated_pages > heap_pages_sorted_length) {
      rb_bug("heap_page_allocate: allocated(%"PRIdSIZE") > sorted(%"PRIdSIZE")
         heap_allocated_pages, heap_pages_sorted_length);
   }

   if (heap_pages_lomem == 0 || heap_pages_lomem > start) heap_pages_lomem = start;
   if (heap_pages_himem < end) heap_pages_himem = end;

   page->start = start;
   page->total_slots = limit;
   page_body->header.page = page;

   for (p = start; p != end; p++) {
      gc_report(3, objspace, "assign_heap_page: %p is added to freelist\n", (void *)p);
      heap_page_add_freeobj(objspace, page, (VALUE)p);
   }

   page->free_slots = limit;

   asan_poison_memory_region(&page->freelist, sizeof(RVALUE*));
   return page;
}

It seems that the rb_aligned_malloc already returns inaccessible pointer:
Heare I am stepping through the rb_aligned_malloc

The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /builddir/build/BUILD/ruby-3.0.3/miniruby
Download failed: No route to host. Continuing without debug info for /lib64/libz.so.1.
Download failed: No route to host. Continuing without debug info for /lib64/libgmp.so.10.
Download failed: No route to host. Continuing without debug info for /lib64/libcrypt.so.2.
Download failed: No route to host. Continuing without debug info for /lib64/libm.so.6.
Download failed: No route to host. Continuing without debug info for /lib64/libc.so.6.
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".

#include "gc.h"
10385 if (end_out_of_range_size > 0) {
10386     if (munmap(aligned + size, end_out_of_range_size)) {
10387         rb_bug("rb_aligned_malloc: munmap failed for end");
10388     }
10389 }
10390 #else
10391     res = (void *)aligned;
10392 #else
10393     char *aligned;
10394     res = malloc(alignment + size + sizeof(void*));
10395     aligned = (char*)res + alignment + sizeof(void*);
10396     aligned -= ((VALUE)aligned & (alignment - 1));
10397     ((void**)aligned)[-1] = res;
10398     res = (void*)aligned;
10399 }
10400
10401 /* alignment must be a power of 2 */
10402 GC_ASSERT(((alignment - 1) & alignment) == 0);
10403 GC_ASSERT(alignment % sizeof(void*) == 0);
10404 return res;
10405 }
10406
10407 static void
10408 rb_aligned_free(void *ptr, size_t size)
10409 {
10410     char *ptr = mmap(NULL, alignment + size, PROT_READ | PROT_WRITE, MAP_PRIVATE | MAP_ANONYMOUS, -1, 0);
10411     if (ptr == MAP_FAILED) {
10412         if (ptr == MAP_FAILED) {
10413             // Error handling...
10414         }
10415     }
10416     char *aligned = ptr + alignment;
10417     p alignment
10418     aligned -= ((VALUE)aligned & (alignment - 1));
10419     p aligned
10420     if (start_out_of_range_size > 0) {
10421         if (munmap(ptr, start_out_of_range_size)) {
10422             // Error handling...
10423         }
10424     }
10425     if (end_out_of_range_size > 0) {
10426         if (munmap(ptr, end_out_of_range_size)) {
10427             // Error handling...
10428         }
10429     }
10430     // Error handling...
10431     // Cleanup...
10432 
10433     // Error handling...
10434     // Cleanup...
10435 
And the originally mapped block is later unmapped, therefore the page_body is not accessible. I have also apply the
6f0b0d1bc558f7a90d77b64d41b98a9c27ddedc7208 but without much success.

#13 - 12/09/2021 02:21 PM - peterzhu2118 (Peter Zhu)

Hmmm, that's really odd. I think I can get access to the ppc64 machine on rubyci.org. I'll try to debug this next week.

#14 - 12/09/2021 03:50 PM - vo.x (Vit Ondruch)
peterzhu2118 (Peter Zhu) wrote in #note-13:

Hmmm, that's really odd. I think I can get access to the ppc64 machine on rubyci.org. I'll try to debug this next week.
If you like, you could ping @sharkcz (Dan Horák) on #fedora-ppc at libera.chat IRC for PPC shell access:

https://fedoraproject.org/wiki/Architectures/PowerPC

#15 - 12/09/2021 03:53 PM - vo.x (Vít Ondruch)

vo.x (Vít Ondruch) wrote in #note-5:

Thanks for looking into this. However, applying these two patches, while fixing i686, it breaks ppc64le :

```
[ 8890/21266] TestGCCompact@test_ast_compacts<internal:gc>:213: [BUG] Couldn't unprotect page 0x0000000140f98000
ruby 3.0.3p157 (2021-11-24 revision 3fb7d2cadc) [powerpc64le-linux]
-- Control frame information ------------------------------------------
Since I have the PPC at hand, here is the full backtrace:

$ make gdb-ruby TESTRUN_SCRIPT=test/ruby/test_gc_compact.rb RUNOPT0='-I.ext/powerpc64le-linux:tool/lib'
revision.h unchanged
Reading symbols from /build/dir/build/BUILD/ruby-3.0.3/ruby... 
warning: File "/build/dir/build/BUILD/ruby-3.0.3/gdbinit" auto-loading has been declined by your 'auto-load safe-path' set to "/debugdir:/datadir/auto-load".
To enable execution of this file add 
add-auto-load-safe-path /build/dir/build/BUILD/ruby-3.0.3/gdbinit 
line to your configuration file "/build/dir/.config/gdb/gdbinit".
To completely disable this security protection add 
set auto-load safe-path / 
line to your configuration file "/build/dir/.config/gdb/gdbinit".
For more information about this security protection see the 
"Auto-loading safe-path" section in the GDB manual. E.g., run from the shell:
info (gdb)auto-loading safe path" 
Function "rb_assert_failure" not defined.
Breakpoint 1 (rb_assert_failure) pending.
Function "rb_bug" not defined.
Breakpoint 2 (rb_bug) pending.
Function "ruby_debug_breakpoint" not defined.
Breakpoint 3 (ruby_debug_breakpoint) pending.
warning: ./breakpoints.gdb: No such file or directory
Download failed: No route to host. Continuing without debug info for /lib64/libc.so.6.
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
Download failed: No such file or directory
Download failed: No route to host. Continuing without debug info for /lib64/libbz.so.1.
Download failed: No route to host. Continuing without debug info for /lib64/libgmp.so.1.
Download failed: No route to host. Continuing without debug info for /lib64/libcrypt.so.2.
Download failed: No route to host. Continuing without debug info for /lib64/libffi.so.6.
Run options: 
--seed=62
# Running tests:

[Detaching after vfork from child process 635]

[1/8] TestGCCompact@test_ast_compacts
Breakpoint 2, 0x00007fff7b0be3c in rb_bug (fmt=0x7fff7df85f8 "Couldn't unprotect page %p") at error.c:768
768 {
    Missing separate debuginfos, use: dnf debuginfo-install glibc-2.34.9000-26.fc36.ppc64le gmp-6.2.1-1.fc36.ppc64le libffi-3.1-28.fc34.ppc64le libxml2-4.4.26-4.fc36.ppc64le zlib-1.2.11-30.fc35.ppc64le 
    (gdb) bt
    #0 0x00007fff7b0be3c in rb_bug (fmt=0x7fff7df85f8 "Couldn't unprotect page %p") at error.c:768
    #1 0x00007fff7de1f50 in unlock_page_body (objspace=optimized out, body=0x10004f4000) at gcc.c:4505
    #2 gc_fill_swept_page (empty_slots=virtual pointer, freed_slots=synthetic pointer, sweep_page=0x100055a90, heap=optimized out, objspace=optimized out) at gcc.c:4780
    #3 gc_page_sweep (sweep_page=optimized out, objspace=optimized out) at gcc.c:4955
    #4 gc_delete (free_pages, objspace=optimized out) at gcc.c:4955
    #5 gc_collect_objects (objspace=optimized out) at gcc.c:4955
    #6 gc_collect (objspace=optimized out) at gcc.c:4955
    #7 gc_collect (objspace=optimized out) at gcc.c:4955
    #8 gc_collect (objspace=optimized out) at gcc.c:4955
    #9 gc_collect (objspace=optimized out) at gcc.c:4955
    #10 gc_collect (objspace=optimized out) at gcc.c:4955
```
#55 invoke_block_from_c_proc (me=0x0, is_lambda=<optimized out>, passed_block_handler=0, kw_splat=0, argv=0x10 0329fd8, argc=0, self=4298951640, proc=<optimized out>, ec=0x100051bc0) at vm.c:1435
#56 vm_invoke_proc (proc=<optimized out>,(argc=<optimized out>, argv=0x100329fd8, kw_splat=<optimized out>, passed_block_handler=0) at vm.c:1464
#57 0x00007fff77dbd8f0 in rb_vm_invoke_proc (ec=<optimized out>, argc=<optimized out>, argv=0x100329fd8, kw_splat=<optimized out>, passed_block_handler=0) at vm.c:1485
#58 0x00007ffff7ba2650 in rb_proc_call (self=<optimized out>, args=<optimized out>) at proc.c:986
#59 0x00007ffff7b9bc18 in rb_call_end_proc (data=4298928280) at eval_jump.c:13
#60 0x00007ffff7b9bb24 in exec_end_procs_chain (procs=procs@entry=0x7ffff7f2a778 <end_procs.lto_priv>, errp=errp@entry=0x100051c38) at eval_jump.c:105
#61 0x00007ffff7ba2650 in rb_proc_call (self=<optimized out>, args=<optimized out>) at proc.c:986
#62 0x00007ffff7b9bb24 in exec_end_procs_chain (procs=procs@entry=0x7ffff7f2a778 <end_procs.lto_priv>, errp=errp@entry=0x100051c38) at eval_jump.c:105
#63 0x00007ffff7ba2650 in rb_call_end_proc (data=4298928280) at eval_jump.c:13
#64 0x00007ffff7b9bc18 in rb_exec_end_proc (ec=ec@entry=0x100051bc0) at eval_jump.c:120
#65 0x00007ffff7ba2650 in rb_exec_end_proc (ec=ec@entry=0x100051bc0) at eval_jump.c:120

(gdb) l
4500 4501 if (!VirtualProtect(body, HEAP_PAGE_SIZE, PAGE_READWRITE, &old_protect)) {
4502 #else
4503 if(mprotect(body, HEAP_PAGE_SIZE, PROT_READ | PROT_WRITE)) {
4504 #endif
4505 rb_bug("Couldn't unprotect page %p", (void *)body);
4506 } else {
4507 gc_report(5, objspace, "Unprotecting page in move %p\n", (void *)body);
4508 }
4509 }

#16 - 12/14/2021 08:40 PM - peterzhu2118 (Peter Zhu)
- File mmap.patch added

I debugged this today. Can you try with commits 0130e17a410d60a10e7041ce98748b8de6946971 and 32b7dfb56a417c1d1c354102351f1825d653bf cherry-picked and then apply the attached patch? I was able to get it working on ppc64.

#17 - 12/15/2021 11:20 AM - vo.x (Vit Ondruch)
peterzhu2118 (Peter Zhu) wrote in #note-16:

I debugged this today. Can you try with commits 0130e17a410d60a10e7041ce98748b8de6946971 and 32b7dfb56a417c1d1c354102351f1825d653bf cherry-picked and then apply the attached patch? I was able to get it working on ppc64.

This is the result:

1) Error:
TestGCCompact#test_ast_compacts:
NotImplementedError: Compaction isn't available on this platform
<internal:gc>:213:in `compact'
/builddir/build/BUILD/ruby-3.0.3/test/ruby/test_gc_compact.rb:146:in `test_ast_compacts'
2) Error:
TestGCCompact#test_compact_count:
NotImplementedError: Compaction isn't available on this platform
<internal:gc>:213:in `compact'
3) Error:
TestGCCompact#test_complex_hash_keys:
NotImplementedError: Compaction isn't available on this platform
<internal:gc>:213:in `verify_compaction_references'
/builddir/build/BUILD/ruby-3.0.3/test/ruby/test_gc_compact.rb:130:in `test_complex_hash_keys'
4) Error:
TestGCCompact#test_gc_compact_stats:
NotImplementedError: Compaction isn't available on this platform
<internal:gc>:213:in `compact'
/builddir/build/BUILD/ruby-3.0.3/test/ruby/test_gc_compact.rb:91:in `test_gc_compact_stats'

Looking at the patch, you might be interested in the following configuration bits:

checking for mmap... yes
checking for sys/user.h... yes
checking whether PAGE_SIZE is compile-time const... no

Comparing to the x86_64 build:
checking whether PAGE_SIZE is compile-time const... yes

#18 - 12/15/2021 02:02 PM - vo.x (Vit Ondruch)
Checking on CentOS Steam 9, it is passing on ppc:

https://kojihub.stream.rdu2.redhat.com/koji/taskinfo?taskID=848077

The configure checks:

cHECKING for mmap... yes
cHECKING for sys/user.h... yes
cHECKING whether PAGE_SIZE is compile-time const... no

#19 - 12/15/2021 02:08 PM - vo.x (Vit Ondruch)
vo.x (Vit Ondruch) wrote in #note-18:

Checking on CentOS Steam 9, it is passing on ppc:

https://kojihub.stream.rdu2.redhat.com/koji/taskinfo?taskID=848077

The configure checks:

CHECKING for mmap... yes
CHECKING for sys/user.h... yes
CHECKING whether PAGE_SIZE is compile-time const... no

Hups, scratch that, it fails also on c9s:

1) Error:
TestGCCompact#test_ast_compacts:
NotImplementedError: Compaction isn't available on this platform
<internal:gc>:213:in `compact'
/builddir/build/BUILD/ruby-3.0.3/test/ruby/test_gc_compact.rb:146:in `test_ast_compacts'

2) Error:
TestGCCompact#test_compact_count:
NotImplementedError: Compaction isn't available on this platform
<internal:gc>:213:in `compact'

3) Error:
TestGCCompact#test_complex_hash_keys:
NotImplementedError: Compaction isn't available on this platform
<internal:gc>:231:in `verify_compaction_references'
/builddir/build/BUILD/ruby-3.0.3/test/ruby/test_gc_compact.rb:130:in `test_complex_hash_keys'

4) Error:
TestGCCompact#test_gc_compact_stats:
NotImplementedError: Compaction isn't available on this platform
<internal:gc>:213:in `compact'
/builddir/build/BUILD/ruby-3.0.3/test/ruby/test_gc_compact.rb:91:in `test_gc_compact_stats'

Not sure why the verbose log does not contain F by the test list ...

#20 - 12/15/2021 02:37 PM - peterzhu2118 (Peter Zhu)
- File mmap.patch added

I am able to repro this. Since ppc64 uses 64KB pages for mmap, we can't use mmap to allocate memory for Ruby pages (since they are 16KB).
Because we can't use mmap, we can't use mprotect used by the read barriers of compaction, so we can't use compaction. I forgot to include an additional change to the patch to skip compaction tests on those systems. Sorry about that. I've attached the new patch.

#21 - 12/16/2021 12:57 PM - vo.x (Vit Ondruch)
Thx, the latest version passes the test suite everywhere and your explanation why it does not work makes sense.

Nevertheless, this makes me wonder in case there are 64 KB pages used for mmap, why Ruby wont use 64 KB pages as well? I assume that the that the 64 KB pages are used not just for mmap but also for other allocations, but I am far from understanding memory pages and what not. Or maybe this is something in works for Ruby 3.1 ...

Sorry for my naive questions and thx for looking into this.

#22 - 12/16/2021 02:44 PM - peterzhu2118 (Peter Zhu)
Thank you for checking the patch! Historically, Ruby has been using 16KB pages, so there's assumptions in the GC about this. This wasn't a problem.
on 64KB page size systems when we were using posix_memalign, but we can no longer use that with compaction (the change from posix_memalign
to mmap was made this year). I will look into allocating pages larger than 16KB so we can use mmap on these platforms.

#23 - 04/21/2022 07:47 AM - vo.x (Vit Ondruch)
@peterzhu2118 (Peter Zhu) I wonder what is the status here. I think you have requested backport of the mmap patch in #18394. However, it seems it have not happened. Was it intentional?

#24 - 04/21/2022 07:58 AM - vo.x (Vit Ondruch)
- Related to Bug #18746: /TestGCCompact/test_(ast_compacts/compact_count/complex_hash_keys/gc_compact_stats)/ fails on PPC added

#25 - 04/23/2022 04:19 PM - peterzhu2118 (Peter Zhu)
I changed the backport status of #18394 for Ruby 3.0 since it doesn't look like the patch was correctly applied.

Files

<table>
<thead>
<tr>
<th>File</th>
<th>Size</th>
<th>Date</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>mmap.patch</td>
<td>9.45 KB</td>
<td>12/14/2021</td>
<td>peterzhu2118 (Peter Zhu)</td>
</tr>
<tr>
<td>mmap.patch</td>
<td>11.1 KB</td>
<td>12/15/2021</td>
<td>peterzhu2118 (Peter Zhu)</td>
</tr>
</tbody>
</table>