Ruby master - Bug #18062
Ruby with enabled LTO segfaults during build
08/05/2021 07:56 AM - vo.x (Vit Ondruch)

Status: Closed
Priority: Normal
Assignee: 
Target version: ruby -v: ruby 3.0.2p107 (2021-07-07 revision 0db68f0233) [x86_64-linux]
Backport: 2.6: UNKNOWN, 2.7: UNKNOWN, 3.0: REQUIRED

Description
After a while, I am trying to enable LTO for Ruby in Fedora. Previously I hit #17052, but now I observe issues much earlier during compilation already:

```
./miniruby -I./lib -I. -I.ext/common ./tool/generic_erb.rb -c -o encdb.h ./template/encdb.h.tmpl
```

```
<internal:gc>:240: [BUG] Segmentation fault at 0x0000000000000004
```

```
After a while, I am trying to enable LTO for Ruby in Fedora. Previously I hit #17052, but now I observe issues much earlier during compilation already:

```
./miniruby -I./lib -I. -I.ext/common ./tool/generic_erb.rb -c -o encdb.h ./template/encdb.h.tmpl
```

```
7f00a75000-7f00a7a77000 rw-p 00031000 fc:05 270939734 /usr/lib64/ld-linux-x86-64.so.2
7ffd63e17000-7ffd64616000 rw-p 00000000 00:00 0 [stack]
7ffd64762000-7ffd64766000 r--p 00000000 00:00 0 [vvar]
7ffd64766000-7ffd64768000 r-xp 00000000 00:00 0 [vdso]
ffffffffffff600000-ffffffffffff601000 r-xp 00000000 00:00 0 [vsyscall]


Associated revisions

Revision a0a8f2ab - 08/11/2021 03:07 AM - nobu (Nobuyoshi Nakada)
Get rid of type-punning pointer casts [Bug #18062]

Revision 573eef7c - 08/13/2021 04:29 AM - nobu (Nobuyoshi Nakada)
Indicate the slow path of Check_Type never return [Bug #18062]

Though this call to rb_check_type is just to raise an exception and never return actually, it can return at least formally. That means a caller function looks like it will access flags even in the special-const cases, and some optimizers may unify the access with the same access just following the call, and re-order it before the guard.

Revision 611da9fa - 08/13/2021 04:29 AM - nobu (Nobuyoshi Nakada)
Mark rb_unexpected_type as "cold" [Bug #18062]

So that it will not interfere the fast path in Check_Type.

History

#1 - 08/05/2021 08:11 AM - vo.x (Vit Ondruch)
GDB output:

$ gdb ./miniruby
GNU gdb (GDB) Fedora 10.2.6.fc35
Copyright (C) 2021 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
Find the GDB manual and other documentation resources online at:

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./miniruby...
(gdb) r -I./lib -I.-ext/common ../tool/generic_erb.rb -c -o encdb.h ./template/encdb.h.tmpl ./enc enc
Starting program: /builddir/build/BUILD/ruby-3.0.2/miniruby -I./lib -I.-ext/common ../tool/generic_erb.rb -c -o encdb.h ./template/encdb.h.tmpl ./enc enc
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/libthread_db.so.1.
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".

Program received signal SIGSEGV, Segmentation fault.
vm_ccs_push_part.0.lto_priv.0 (klass=93824996623120, ccs=0x0, ci=0x22f100140003, cc=0x5555555595ddfd0) at /builddir/build/BUILD/ruby-3.0.2/vm_insnhelper.c:1579
1579 if (LIKELY(ccs->len == ccs->capa))
Missing separate debuginfos, use: dnf debuginfo-install glibc-2.34-1.fc35.x86_64 gmp-6.2.0-7.fc35.x86_64 libxml2-4.4.23-2.fc35.x86_64 zlib-1.2.11-30.fc35.x86_64
(gdb) where
#0 0x7ffff7fc9000

09/16/2021
#1 0x00005555557eb7e in vm_ccs_push (cc=0x00005555555ddf0, ci=0x22f100140003, ccs=optimized out), klass=938249 96623120) at /include/ruby/internal/fl_type.h:237
#2 vm_search_cc (klass=entry=93824996623120, ci=0x22f100140003) at /buildidl/build/BUILD/ruby-3.0.2/vm_ insnhelper.c:1718
#3 0x00005555557ed30 in rb_vm_search_method_slowpath (klass=93824996623120, ci=optimized out) at /buildidl/build/BUILD/ruby-3.0.2/vm_insnhelper.c:1736
#4 vm_search_method_slowpath (cd_owner=93824996623120, cd=entry=0x5555555a358f0, klass=93824996623120) at /buildidl/build/BUILD/ruby-3.0.2/vm_insnhelper.c:1758
#5 0x00005555557af4 in vm_search_method_fastpath (klass=optimized out), cd=0x5555555a358f0, cd_owner=optimized out) at /buildidl/build/BUILD/ruby-3.0.2/vm_insnhelper.c:1824
#6 vm_sendish (ec=0x5555555909c80, reg_cfp=0x7ffff7f84f68, cd=0x5555555a358f0, block_handler=optimized out), method_explore=optimized out) at /buildidl/build/BUILD/ruby-3.0.2/vm_insnhelper.c:4526
#7 0x0000555555795b8 in vm_exec_core (ec=0x5555555909c80, initial=0) at /buildidl/build/BUILD/ruby-3.0.2/insn.s:789
#8 0x00005555557c0ea0 in rb_vm_exec (ec=0x5555555909c80, mjit_enable_p=optimized out) at /buildidl/build/BUILD/ruby-3.0.2/vm.c:2172
#9 0x00005555557a5ff4 in vm_search_method_fastpath (klass=optimized out), cd=0x5555555a358f0, cd_owner=optimized out) at /buildidl/build/BUILD/ruby-3.0.2/vm_insnhelper.c:1736
#10 Init_builtin_gc () at /buildidl/build/BUILD/ruby-3.0.2/gc.rbinc:212
#11 Init_builtin_gc () at /buildidl/build/BUILD/ruby-3.0.2/inits.c:88
#12 ruby_opt_init (opt=0x7fffffffdfb0) at /buildidl/build/BUILD/ruby-3.0.2/ruby.c:1506
#13 0x000055555552e6e2 in rb_load_with_builtin_functions (table=0x55555550240 <gc_table>, feature_name=0x5555 5588e03 "gc") at /buildidl/build/BUILD/ruby-3.0.2/minibuiltin.c:48
#14 Init_builtin_gc () at /buildidl/build/BUILD/ruby-3.0.2/ruby.c:1506
#15 0x0000555555572e6e2 in load_file (argp_v=140737488309936) at /buildidl/build/BUILD/ruby-3.0.2/ruby.c:2181
#16 0x00005555555600456 in rb_load (b_proc=0x55555555e230 <load_file_internal>, data=140737488309936, e_proc= optimized out), data2=optimized out) at /buildidl/build/BUILD/ruby-3.0.2/eval.c:1162
#17 0x000055555557aa65 in load_file (opt=0x7fffffffdfb0, script=1, f=optimized out), f_name=optimized out), parser=938249964989920) at /buildidl/build/BUILD/ruby-3.0.2/ruby.c:2323
#18 process_options (argc=6, argv=0x7fffffffde350, opt=0x7fffffffdfb0) at /buildidl/build/BUILD/ruby-3.0.2/eval.c:138
#19 0x0000555555509de9 in ruby_process_options (argc=11, argv=0x7fffffffde328) at /buildidl/build/BUILD/ruby-3. 0.2/ruby.c:230
#20 0x00005555555811e7 in main (argc=optimized out), argv=optimized out) at ./main.c:50 (gdb) list
1574 } 
1575 } else if (! vm_ci_markable(ci)) { 
1576 return; 
1577 
1578 if (UNLIKELY(ccs->len == ccs->capa)) { 
1579 if (ccs->capa == 0) { 
1580 ccs->capa = 1; 
1581 ccs->entries = ALLOC_N(struct rb_class_cc_entries_entry, ccs->capa); 
1582 } 
1583 } 
(gdb) 
1584 else { 
1585 ccs->capa *= 2; 
1586 REALLOC_N(ccs->entries, struct rb_class_cc_entries_entry, ccs->capa); 
1587 
1588 } 
1589 VM_ASSUME(ccs->len < ccs->capa); 
1590 const int pos = ccs->len++; 
1591 RB_OBJ_WRITE(klass, &ccs->entries[pos].ci, ci); 
1592 RB_OBJ_WRITE(klass, &ccs->entries[pos].cc, cc); 
(gdb)

#2 - 08/05/2021 08:30 AM - vo.x (Vit Ondruch)
I'd say it goes sideways somewhere here:

https://github.com/ruby/ruby/blob/0db68f023372b634603c741ca94588b457be084c/vm_insnhelper.c#L1708

I'm not sure why the rb_id_table_lookup does not provide any value for css, but there is no check for NULL further down the path.

#3 - 08/05/2021 08:42 AM - vo.x (Vit Ondruch)
vo.x (Vit Ondruch) wrote in #note-2:

I'd say it goes sideways somewhere here:

https://github.com/ruby/ruby/blob/0db68f023372b634603c741ca94588b457be084c/vm_insnhelper.c#L1708
I'm not sure why the rb_id_table_lookup does not provide any value for css, but there is no check for NULL further down the path.

Actually, this is weird:

Breakpoint 1, vm_ccs_push (cc=0x55555595ddf0, ci=0x22f100140003, css=0x555555a31f80, klass=93824996623120) at ./include/ruby/internal/fl_type.h:237
237 return RBASIC(obj)->flags & flags;
(gdb) s
Breakpoint 1, vm_ccs_push.part.0.lto_priv.0 (klass=93824996623120, css=0x0, ci=0x22f100140003, css=0x55555595ddf0) at /build/build/RUBY-3.0.2/vm_insnhelper.c:1570
1570 vm_ccs_push(VALUE klass, struct rb_class_cc_entries *ccs, const struct rb_callinfo *ci, const struct rb_cal...)

It seems the ccs has some non NULL value initially, but it changes into NULL with the next step ...

#4 - 08/05/2021 11:32 AM - nobu (Nobuyoshi Nakada)
- File 0001-Get-rid-of-type-punning-pointer-cast.patch added

Maybe a strict aliasing issue?

#5 - 08/06/2021 10:22 AM - vo.x (Vit Ondruch)
nobu (Nobuyoshi Nakada) wrote in #note-4:

Maybe a strict aliasing issue?

The patch changes the error:

```
./miniruby -I./lib -I. -I.ext/common ./tool/generic.erb.rb -c -o encdb.h ./template/encdb.h.tmpl ./enc enc
ruby 3.0.2p107 (2021-07-07 revision 0db68f0233) [x86_64-linux]
```

```
== Control frame information ==-------------------------------
c:0003 p:0046 s:0013 e:000008 CLASS <internal:ractor>:431
c:0002 p:0007 s:0006 e:000005 TOP <internal:ractor>:228 [FINISH]
c:0001 p:0000 s:0003 E:001c30 (none) [FINISH]
== Ruby level backtrace information ==-------------------------
<internal:ractor>:228:in `<internal:ractor>'
<internal:ractor>:431:in `<class:Ractor>'
== Machine register context ==-------------------------------
RIP: 0x0000559b9c12eb82 RBP: 0x0000559b9cefc210 RSP: 0x0000559b9ce55d0
RAX: 0x0000000000000001 RBX: 0x000000000000009f RCX: 0x00005fa0da7cb060
RDX: 0x0000000000000003 RDI: 0x0000559b9cfe4650 RSI: 0x000000000000009f
R8: 0x0000000000000000 R9: 0x0000559b9ce0a130 R10: 0x0000559b9ce0a130
R11: 0x0000559b9c9b6a10 R12: 0x0000559b9b010007 R13: 0x0000559b9c3d5e0
R14: 0x0000559b9c6d0d80 R15: 0x0000559b9c3d658 EFL: 0x0000000000010202
== C level backtrace information ==---------------------------
<miniruby.rb_print_backtrace+0x19> [x5559b9c154c49]
<miniruby.rb_vm_bugreport+0x215> [x5559b9c158aa5]
<miniruby.rb_bug_for_fatal_signal+0x4f4> [x0559b9bf8bfe4]
<miniruby.(sigsegv+0x4d) [x0559b9c0be14d]
<miniruby.(rs_insn+0x10) [x0559b9c4e51f040]
```

* Loaded script: ./miniruby
I should probably include a few more lines from the log which include the build options:

```
```

I've missed here.

```
vo.x (Vit Ondruch) wrote in #note-5:

```
1641     if (rb_id_table_lookup(cc_tbl, mid, (VALUE *)&ccs)) {
1642          const int ccs_len = ccs->len;
1643          VM_ASSERT(vm_ccs_verify(ccs, ccs, mid, class));
1644          if (UNLIKELY(METHOD_ENTRY_INVALIDATED(ccs->cme))) {
1645              rb_vm_ccs_free(ccs);
1646     }
```

I've missed here.

```n
vo.x (Vit Ondruch) wrote in #note-5:

```
1641     if (rb_id_table_lookup(cc_tbl, mid, (VALUE *)&ccs)) {
1642          const int ccs_len = ccs->len;
```

Thx, this helps. However, there is more:

```
I've missed here.
```

```n
vo.x (Vit Ondruch) wrote in #note-5:

```
1641     if (rb_id_table_lookup(cc_tbl, mid, (VALUE *)&ccs)) {
1642          const int ccs_len = ccs->len;
```

I've missed here.

```n
vo.x (Vit Ondruch) wrote in #note-5:

```
1641     if (rb_id_table_lookup(cc_tbl, mid, (VALUE *)&ccs)) {
1642          const int ccs_len = ccs->len;
```

I've missed here.
0x00007fff7e6732 in vm_call_cfunc_with_frame (ec=0x55555555de90, reg_cfp=0x7ffffff7efef30, calling=<optimized out>) at /buildid/build/BUILD/ruby-3.0.2/vm_insnhelper.c:2929
0x00007fff7e6a356 in vm_sendish (ec=0x55555555de90, reg_cfp=0x7ffffff7efef30, cd=0x55555556b6d0, block_handler=<optimized out>, method_explorer=<optimized out>) at /buildid/build/BUILD/ruby-3.0.2/vm_callinfo.h:336
0x00007ffffff7d9998 in require_internal (ec=0x55555555de90, fname=<optimized out>, exception=1) at /buildid/build/BUILD/ruby-3.0.2/load.c:1065
0x00007ffff7e6eb48 in vm_exec_core (ec=0x55555555de90, initial=2065) at /buildid/build/BUILD/ruby-3.0.2/insns.def:789
0x00007ffff7e874c0 in rb_vm_exec (ec=0x55555555de90, mjit_enable_p=<optimized out>) at /buildid/build/BUILD/ruby-3.0.2/vm.c:2172
0x00007ffff7ceb06f in rb_ec_exec_node (ec=ec@entry=0x55555555de90, n=n@entry=0x55555559a790) at /buildid/build/BUILD/ruby-3.0.2/eval.c:317
0x00007ffffff7ebf18 in ruby_run_node (n=0x555555559a790) at /buildid/build/BUILD/ruby-3.0.2/eval.c:375

#10 - 08/07/2021 02:01 AM - nobu (Nobuyoshi Nakada)
OK, how about this?
https://github.com/ruby/ruby/pull/4716

#11 - 08/09/2021 09:44 AM - vo.x (Vit Ondruch)
- File 4716.patch added
Testing with Ruby 3.0.2, I had to modify the patch slightly and I was able to build the Ruby as well as pass test suite. However, I'd like if you can review the modifications I did for Ruby 3.0.2 in the vm_method.c hunks (I have replaced the hunk #4 with hunk #5).

#12 - 08/09/2021 10:23 AM - vo.x (Vit Ondruch)
Unfortunately, testing on other arches, the #17052 is still an issue :( #17052

#13 - 08/10/2021 02:31 AM - nobu (Nobuyoshi Nakada)
That patch seems fine, but maybe f4ce78d5c139 and 9c769575bfa2 also need to be backported. #17553

#14 - 08/10/2021 08:54 AM - vo.x (Vit Ondruch)
- Backport changed from 2.6: UNKNOWN, 2.7: UNKNOWN, 3.0: UNKNOWN to 2.6: UNKNOWN, 2.7: UNKNOWN, 3.0: REQUIRED

nobu (Nobuyoshi Nakada) wrote in #note-13:

That patch seems fine

Thx for confirmation

but maybe f4ce78d5c139 and 9c769575bfa2 also need to be backported.

If would definitely make my life easier, but nobody have requested the backport of #17553 yet and I have not met the issue myself. For the moment, I'm going to set the backport flag for this ticket and let nagachika (Tomoyuki Chikanaga) to decide.
I still have to figure out if I am going to use this downstream or not.

#15 - 08/10/2021 06:33 PM - jaruga (Jun Aruga)
- File jaruga_make_check.log added
- File jaruga_make_cflags_with_O2_lto_flags.log added

I identified which compiler flags make this error, segmentation fault. I tested it on the latest master 28d03ee776af4d630556fcaad7305340cbbf9cde. I think this info helps for developers to reproduce this error on their machine.

My testing environment is Fedora 34 x86_64 and the used gcc version is as follows.

```bash
$ gcc --version
gcc (GCC) 11.2.1 20210728 (Red Hat 11.2.1-1)
Copyright (C) 2021 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

As a note, on Fedora, the following cflags is set as a default for both gcc and clang.
https://src.fedoraproject.org/rpms/redhat-rpm-config/blob/rawhide/f/macros#_341

```bash
%_gcc_lto_cflags  -flto=auto  -ffat-lto-objects
%_clang_lto_cflags -flto
...
%_general_options  -O2 %(?_lto_cflags) -fexceptions -g -grecord-gcc-switches -pipe
```

CFLAGS=''-flto=auto  -ffat-lto-objects' => make: ok

First, I confirmed that with CFLAGS=''-flto=auto  -ffat-lto-objects', the make command works and finishes without error against my expectation.

```bash
$ git clean -fdx
$ autoconf
$ CFLAGS=''-flto=auto  -ffat-lto-objects' ./configure --enable-shared --prefix $(pwd)/dest
$ make 2>&1 | tee make.log
```

**CFLAGS=''-O2  -flto=auto  -ffat-lto-objects' => make: error**

After spending some time, I found the CFLAGS=''-O2  -flto=auto  -ffat-lto-objects' (-O2 + lto flags) causes the make error. I would attach the log as jaruga_make_cflags_with_O2_lto_flags.log.

```bash
$ git clean -fdx
$ autoconf
$ CFLAGS=''-O2 -flto=auto -ffat-lto-objects' \  ./configure --enable-shared --prefix $(pwd)/dest
$ make 2>&1 | tee make.log
... + cp enc/jis/props.h.blt enc/jis/props.h
generating encdb.h
<internal:gc>:240: [BUG] Segmentation fault at 0x0000000000000004
rub y 3.1.0dev (2021-08-10T10:08:38Z master 28d03ee776) [x86_64-linux]
... => error
```

**CFLAGS=''-O2  -flto=auto  -ffat-lto-objects' + a commit on the PR [https://github.com/ruby/ruby/pull/4716] => make: ok, make check: error**

On the latest master 28d03ee776af4d630556fcaad7305340cbbf9cde + applying a commit on the PR [https://github.com/ruby/ruby/pull/4716], the make is ok, but make check still has error. I would attach the log as jaruga_make_check.log.

```bash
$ git clean -fdx
$ autoconf
$ CFLAGS=''-O2 -flto=auto -ffat-lto-objects' \  ./configure --enable-shared --prefix $(pwd)/dest
$ make 2>&1 | tee make.log
$ make check 2>&1 | tee make_check.log
... Run options:-
  --seed=91697
  "--ruby=./miniruby -I.lib -I.common ./tool/runruby.rb --extout=.ext -- --disable-gems"
  --excludes-dir=./test/excludes
  --name=/memory_leak/
# Running tests:
Leaked file descriptor: DRbTests::TestDRbTCP#test_immediate_close: 12 : #<TCPSocket:fd 12, AF_INET6, ::1, 3885
Leaked file descriptor: DRbTests::TestDRbTCP#test_immediate_close: 13 : #<IO:fd 13>
Leaked file descriptor: PPTestModule::PPFileStatTest#test_nothing_raised: 7 #<File::Stat dev=0xfd02, ino=13370
094, mode=0100664, nlink=1, uid=0, gid=0, rdev=0x80, size=9253600, blksize=4096, blocks=18080, atime=2021-08-10

COMMAND   PID   USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
ruby 1158696 jaruga  7r   REG 2532  9253600 13370094 /var/lib/sss/mc/passwd
/home/jaruga/var/git/ruby/ruby/test/ruby/test_hash.rb:2080: [BUG] Segmentation fault at 0x0000000000000003
ruby 3.1.0dev (2021-08-10T17:41:00Z master 5340724165) [x86_64-linux]
=> error
#16 - 08/11/2021 03:08 AM - nobu (Nobuyoshi Nakada)
- Status changed from Open to Closed

Applied in changeset git|a0a8f2abf533702b2cd96e79f700ce5b9cd94f50.

Get rid of type-punning pointer casts [Bug #18062]

#17 - 08/11/2021 03:10 AM - nobu (Nobuyoshi Nakada)
- Status changed from Closed to Open

As https://github.com/ruby/ruby/pull/4716 itself seems to make things better at least, I merged it for now.

#18 - 08/11/2021 07:42 AM - nobu (Nobuyoshi Nakada)
I could reproduce the segfault with the master c59da370a5d, just by Hash.ruby2_keywords_hash?.(1).

* thread #1, queue = 'com.apple.main-thread', stop reason = EXC_BAD_ACCESS (code=1, address=0x3)
frame #0: 0x00000001000c82f0 miniruby`rb_hash_s_ruby2_keywords_hash_p.lto_priv.0
miniruby`rb_hash_s_ruby2_keywords_hash_p.lto_priv.0:
  0x1000c82f0 <+0>: movq (%rsi), %rax
  0x1000c82f3 <+3>: movq %rsi, %rdi
  0x1000c82f6 <+6>: andl $0x7, %esi
  0x1000c82f9 <+9>: jne 0x1002b6683 ; rb_hash_s_ruby2_keywords_hash_p.lto_priv.0.cold
Target 0: (miniruby) stopped.
(1ldb) disassemble
miniruby`rb_hash_s_ruby2_keywords_hash_p.lto_priv.0:
  0x1000c82f0 <+0>: movq (%rsi), %rax
  0x1000c82f3 <+3>: movq %rsi, %rdi
  0x1000c82f6 <+6>: andl $0x7, %esi
  0x1000c82f9 <+9>: jne 0x1002b6683 ; rb_hash_s_ruby2_keywords_hash_p.lto_priv.0.cold
  0x1000c82ff <+15>: testq $-0x9, %rdx
  0x1000c8306 <+22>: je 0x1002b6683 ; rb_hash_s_ruby2_keywords_hash_p.lto_priv.0.cold
  0x1000c830c <+28>: movq %rax, %rdx
  0x1000c830f <+31>: andl $0x1f, %edx
  0x1000c8312 <+34>: cmpq %dx, %rdx
  0x1000c8316 <+38>: jne 0x1002b6683 ; rb_hash_s_ruby2_keywords_hash_p.lto_priv.0.cold
  0x1000c831c <+44>: andl $0x2000, %eax
  0x1000c8321 <+49>: movl %edx
  0x1000c8326 <+54>: cmovneq %rdx, %rax
  0x1000c832a <+58>: retq
  0x1000c832c <+59>: nop1 (trax,trax)
(1ldb) p $rsi
(unsigned long) $0 = 3

Seems rbimpl_RB_TYPE_P.fastpath is re-ordered, and RB_BUILTIN_TYPE(obj) is fetched before RB_SPECIAL_CONST_P(obj).

```ruby
else if (RB_SPECIAL_CONST_P(obj)) { return false; }
else if (t == RB_BUILTIN_TYPE(obj)) { return true; }
```

#19 - 09/11/2021 01:09 PM - vo.x (Vit Ondruch)
Just curious, would it make sense to update the Ruby CI to test LTO? Maybe even by default? Would there be any downside?
Just curious, would it make sense to update the Ruby CI to test LTO? Maybe even by default? Would there be any downside?

Yes, I think it makes sense to me, and the `.github/workflows/compilers.yml` is a right place to add. Actually I have been trying to add the cases like this on a branch on forked repo on the yesterday's latest master 280d03ee776a4d630568caa7305340cbcf9ode. You can see the CI result. The behavior is a little different. The make is ok against my expectation. But it's dead in the process of make test with wrong result "green". Now I am trying to reproduce it on Ruby CI's Fedora 32 server (it's the latest version Fedora server on Ruby CI servers) for developer's convenience.

Fiber count: 10000 (skipping)
PASS all 1417 tests

For detail of the current compilers.yml matrix syntax's limitation, we can only set a kind of environment variable. E.g. we can not set both default_cc and optflags variables for a case of the matrix. I am trying to change the limitation too for a better implementation.

diff --git a/.github/workflows/compilers.yml b/.github/workflows/compilers.yml
index af0be21272..31112c79bf 100644
--- a/.github/workflows/compilers.yml
+++ b/.github/workflows/compilers.yml
@@ -55,6 +55,7 @@ jobs:
-  { key: default_cc, name: gcc-6, value: gcc-6, container: gcc-6 }
-  { key: default_cc, name: gcc-5, value: gcc-5, container: gcc-5 }
-  { key: default_cc, name: gcc-4.8, value: gcc-4.8, container: gcc-4.8 }
-  { key: default_cc, name: clang-14, value: clang-14, container: clang-14 }
-  { key: default_cc, name: clang-13, value: clang-13, container: clang-13 }
-  { key: default_cc, name: clang-12, value: clang-12, container: clang-12 }
+  { key: default_cc, name: clang-14, value: clang-14, container: clang-14 }
  { key: default_cc, name: clang-13, value: clang-13, container: clang-13 }
  { key: default_cc, name: clang-12, value: clang-12, container: clang-12 }
@@ -67,6 +68,7 @@ jobs:
-  { key: default_cc, name: clang-5.0, value: clang-5.0, container: clang-5.0 }
-  { key: default_cc, name: clang-4.0, value: clang-4.0, container: clang-4.0 }
-  { key: default_cc, name: clang-3.9, value: clang-3.9, container: clang-3.9 }
+  { key: default_cc, name: 'clang-14 -O2 -flto=auto', value: 'clang-14 -O2 -flto=auto', container: clang-14 }
  { key: crosshost, name: aarch64-linux-gnu, value: aarch64-linux-gnu, container: crossbuild-essential-arm64 }

#21 - 08/11/2021 03:03 PM - nobu (Nobuyoshi Nakada)

Probably a memory barrier would work.

diff --git a/include/ruby/internal/value_type.h b/include/ruby/internal/value_type.h
index 52b284cb4a2..f6231d2dcac 100644
--- a/include/ruby/internal/value_type.h
+++ b/include/ruby/internal/value_type.h
@@ -156,6 +156,11 @@
 RB_BUILTIN_TYPE(VALUE obj)
 {
   RBIMPL_ASSERT_OR_ASSUME(! RB_SPECIAL_CONST_P(obj));
+  if defined __GNUC__ && !defined __clang__
+  /* Don't move the access to `flags` before the preceding
+     * RB_SPECIAL_CONST_P check. */
+  __asm volatile("": : "memory");
+  #endif
+  VALUE ret = RBASIC(obj)->flags & RUBY_T_MASK;
+  return RBIMPL_CAST((enum ruby_value_type)ret);
 } return;}
@@ -351,6 +356,7 @@ Check_Type(VALUE v, enum ruby_value_type t)
 slowpath: /* <- :TODO: mark this label as cold. */
 rb_check_type(v, t);
 + RBIMPL_UNREACHABLE();

#22 - 08/11/2021 03:05 PM - jaruga (Jun Aruga)
But it's dead in the process of make test with wrong result "green".

Sorry, my mistake. The make test just finished without error on the Ubuntu container. So, the "green" is correct. The steps after make install were skipped due to the conditional execution for the CI result.

#23 - 08/11/2021 04:52 PM - jaruga (Jun Aruga)

nobu (Nobuyoshi Nakada) wrote in #note-21:

Probably a memory barrier would work.

I confirmed that the make and make check work with the patch applied on the latest master 79cc566ab4c9df75f125ecf413a27d353a9756c08 on my local Fedora 34 (gcc 11.2.1-1) with CFLAGS="-O2 -flto=auto -ffat-lto-objects" by following commands.

git clean -f dx
autoconf
CFLAGS="-O2 -flto=auto -ffat-lto-objects"
./configure --enable-shared --prefix $(pwd)/dest
make 2>&1 | tee make.log
make check 2>&1 | tee make_check.log

However with the GitHub Actions CI where I added the new cases as I wrote above. Both the gcc/clang cases still fail after applying the patch.

Before applying the patch.
https://github.com/junaruga/ruby/runs/3302937317?check_suite_focus=true#step:16:126

After applying the patch.
https://github.com/junaruga/ruby/runs/3303099400?check_suite_focus=true#step:16:226

#24 - 08/11/2021 05:02 PM - vo.x (Vit Ondruch)

So I have tried to get more details about the #17052:

$ gdb --args ./miniruby -e 'Process.kill("SIGSEGV", $$)'
GNU gdb (GDB) Fedora 10.2-6.fc35
Copyright (C) 2021 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "ppc64le-redhat-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
Find the GDB manual and other documentation resources online at:

For help, type "help".
Type "apropos word" to search for commands related to "word"
...Reading symbols from ./miniruby...
warning: File "/builddir/build/BUILD/ruby-3.0.2/.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-3.0.2/.gdbinit
line to your configuration file "/builddir/.gdbinit".
To completely disable this security protection add
  set auto-load safe-path / line to your configuration file "/builddir/.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-3.0.2/miniruby -e 'Process.kill("SIGSEGV", $$)'
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.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/libsm.so.6.
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".

Program received signal SIGSEGV, Segmentation fault.
0x0000000000000000 in kill () from /lib64/libc.so.6
Missing separate debuginfos, use: dnf debuginfo-install glibc-2.34-1.fc35.ppc64le gmp-6.2.0-7.fc35.ppc64le libxcrypt-4.4.24-1.fc35.ppc64le xlib-1.2.11-30.fc35.ppc64le
(gdb) c
Continuing.
-e1: [BUG] Segmentation fault at 0x590fb15c0000001f
ruby 3.0.2p107 (2021-07-07 revision 0dbf80233) [powerpc64le-linux]

-- Control frame information ---------------------------------------------
c:0003 p:---- s:0012 e:000011 CFUNC :kill

-- Ruby level backtrace information --------------------------------------
-e1: in '<main>'
-e1: in 'kill'

-- C level backtrace information ----------------------------------------
Program received signal SIGSEGV, Segmentation fault.
 0x000000010031fa44 in uleb128 (p=0x1005986a0) at addr2line.c:200  
 200 unsigned char b = *(unsigned char *)(*p)++;

(gdb) bt
#0  0x000000010031fa44 in uleb128 (p=0x1005986a0) at addr2line.c:200
#1 dl_read_die (reader=0x100598668, die=0x100598588) at addr2line.c:1335
#2 0x0000000100337d4c in read_abstract_origin (line=0x1005985f8, abstract_origin=760312279580551, reader=0x10
#3 debug_info_read (traces=<optimized out>, offset=<optimized out>, lines=<optimized out>, num_traces=<optimi
#4 fill_lines (num_traces=num_traces@entry=19, check_debuglink=check_debuglink@entry=1, objp=0x1005990a0, lines=lines@entry=0x1005e2c50, offset=0, offset=entry=-1,
#5 0x0000000100338d1c in rb_dump_backtrace_with_lines.constprop.0 (num_traces=<optimized out>, traces=<optimi
#6 0x000000010031d1f0 in rb_dump_backtrace_inliner (traces=<optimized out>, constprop=constprop.0) at vm_dump.c:2287
#7 0x0000000100332df8 in _addr2line (line=0x1005985f8, lines=lines@entry=0x1005e2c50, offset=0, offset=entry=-1,
#8 0x0000000100332dfc in _addr2line (line=0x1005985f8, lines=lines@entry=0x1005e2c50, offset=0, offset=entry=-1,
#9 0x000000010031d1f0 in rb_dump_backtrace_inliner (traces=<optimized out>, constprop=constprop.0) at vm_dump.c:2287

#10 <signal handler called>
#11 0x000000010031d1f0 in _addr2line (line=0x1005985f8, lines=lines@entry=0x1005e2c50, offset=0, offset=entry=-1,
#12 0x000000010031d1f0 in rb_dump_backtrace_inliner (traces=<optimized out>, constprop=constprop.0) at vm_dump.c:2287
#13 0x000000010031d1f0 in rb_dump_backtrace_inliner (traces=<optimized out>, constprop=constprop.0) at vm_dump.c:2287
#14 0x000000010031d1f0 in rb_dump_backtrace_inliner (traces=<optimized out>, constprop=constprop.0) at vm_dump.c:2287
#15 0x000000010031d1f0 in rb_dump_backtrace_inliner (traces=<optimized out>, constprop=constprop.0) at vm_dump.c:2287
#16 0x000000010031d1f0 in rb_dump_backtrace_inliner (traces=<optimized out>, constprop=constprop.0) at vm_dump.c:2287
#17 0x000000010031d1f0 in rb_dump_backtrace_inliner (traces=<optimized out>, constprop=constprop.0) at vm_dump.c:2287
#18 0x000000010031d1f0 in rb_dump_backtrace_inliner (traces=<optimized out>, constprop=constprop.0) at vm_dump.c:2287
#19 0x000000010031d1f0 in rb_dump_backtrace_inliner (traces=<optimized out>, constprop=constprop.0) at vm_dump.c:2287
#20 0x000000010031d1f0 in rb_dump_backtrace_inliner (traces=<optimized out>, constprop=constprop.0) at vm_dump.c:2287
#21 0x000000010031d1f0 in rb_dump_backtrace_inliner (traces=<optimized out>, constprop=constprop.0) at vm_dump.c:2287
#22 0x000000010031d1f0 in rb_dump_backtrace_inliner (traces=<optimized out>, constprop=constprop.0) at vm_dump.c:2287

#25 - 08/11/2021 06:07 PM - xtkoba (Tee KOBAYASHI)
I would humbly advise that the -fno-strict-aliasing flag be appended to CFLAGS by default. See also #17540.

#26 - 08/12/2021 08:18 AM - vo.x (Vít Ondruch)
xtkoba (Tee KOBAYASHI) wrote in #note-25:

I would humbly advise that the -fno-strict-aliasing flag be appended to CFLAGS by default. See also #17540.

I have not find any guidelines about/against -fno-strict-aliasing in Fedora, so from that POV it would be acceptable. But I would like to avoid to have this as a downstream only option. Therefore if this should be the solution, I'd prefer if this option is applied upstream.

BTW, In your experience, do you think trying something like [1] could help catch such issues earlier?

[1] https://gist.github.com/shafik/848ae25ee209f698763cf8f2272a5f8f#catching-strict-aliasing-violations
BTW, In your experience, do you think trying something like [1] could help catch such issues earlier?

I'm sorry but I'm not familiar with techniques for detecting strict aliasing violation. It would be nice if we had some sort of sanitizer for that. It seems there was once a project to implement one [2], which is not heard now.


I was able to reduce the Hash.ruby2_keywords_hash?(1) repro and report it to GCC. It looks like we are hitting an optimizer bug with LTO: https://gcc.gnu.org/bugzilla/show_bug.cgi?id=101868

For workarounds, putting __attribute__((optimize(1))) on functions that experience this might be an option.

Just testing the issue from comment 24 and there is not difference with or without -fno-strict-aliasing

The problem described in comment 24 does not look like very much related to the other part. Maybe worth reopening #17052?

- Status changed from Open to Closed

Indicate the slow path of Check_Type never return [Bug #18062]

Though this call to rb_check_type is just to raise an exception and never return actually, it can return at least formally. That means a caller function looks like it will access flags even in the special-const cases, and some optimizers may unify the access with the same access just following the call, and re-order it before the guard.

I noticed gcc-11 LTO and clang-14 LTO cases are added to the CI. Thanks nobu (Nobuyoshi Nakada)! https://github.com/ruby/ruby/commit/29dd70492eb3e3f0018c3f0018b0f3f4ae7702a208f https://github.com/ruby/ruby/commit/24416d51c58364b8b9817e07983ae651272b033

Files

<table>
<thead>
<tr>
<th>Patch File</th>
<th>Size</th>
<th>Modification Date</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001-Get-rid-of-type-punning-pointer-cast.patch</td>
<td>1.41 KB</td>
<td>08/06/2021</td>
<td>nobu (Nobuyoshi Nakada)</td>
</tr>
<tr>
<td>4716.patch</td>
<td>7.33 KB</td>
<td>08/09/2021</td>
<td>vo.x (Vit Ondruch)</td>
</tr>
<tr>
<td>jaruga_make_cflags_with_O2_lto_flags.log</td>
<td>16.2 KB</td>
<td>08/10/2021</td>
<td>jaruga (Jun Aruga)</td>
</tr>
<tr>
<td>jaruga_make_check.log</td>
<td>263 KB</td>
<td>08/10/2021</td>
<td>jaruga (Jun Aruga)</td>
</tr>
</tbody>
</table>