Ruby master - Bug #13758

TestRubyOptions#test_segv_setproctitle segfaults on AARCH64

07/20/2017 02:56 PM - vo.x (Vit Ondruch)

Status: Feedback
Priority: Normal
Assignee: 
Target version: 
ruby -v: 
  ruby -v: ruby 2.5.0dev (2017-07-20 trunk 59376) [aarch64-linux]
Backport: 2.2: UNKNOWN, 2.3: UNKNOWN, 2.4: UNKNOWN

Description

Or may be does not segfault properly?

1) Failure:
TestRubyOptions#test_segv_setproctitle 

1. [2/2] Assertion for "stderr"
   | Expected /
   | \[NOTE]\n   | You\smay\shave\sencountered\sa\sbug\sin\sthe\sRuby\sinterpreter\sor\sexten
tion\slibraries.
   | Bug\reports\sare\swelcome.\n   | (?!.*n)?
   | For\sdetails:\shttp://\.\*.\ruby-lang.org/\.\*
   | \n   | (?:
   | \[IMPORTANT]\n   | (?!.+n)+
   | \n   | )?
   | /x

to match
| "n*+
| "-- Ruby level backtrace information
| "-- C level backtrace information
| */build\dir/build/\BUI\LD/ruby-2.5.0-r59376/libruby.so.2.5.0 (rb_print_backtrace+0x20) [0xffff940b570] vm_dump.c:671\n*+
| */build\dir/build/\BUI\LD/ruby-2.5.0-r59376/libruby.so.2.5.0 (rb_vm_bugreport+0x8c) [0xffff940b5b0c] vm_dump.c:941\n*+
| */build\dir/build/\BUI\LD/ruby-2.5.0-r59376/libruby.so.2.5.0 (rb_bug_context+0xb8) [0xffff93f92528] error.c:534\n*+
| */build\dir/build/\BUI\LD/ruby-2.5.0-r59376/libruby.so.2.5.0 (sigsegv+0x4c) [0xffff940c2e4] signal.c:930\n*+
| linux-vdso.so.1 [0xffff942096c0]\n*+
| [0xffff93c230c8]\n*+
| */build\dir/build/\BUI\LD/ruby-2.5.0-r59376/libruby.so.2.5.0 (rb_fKill+0x2c4) [0xffff940d1bc] signal.c:498\n*+
| */build\dir/build/\BUI\LD/ruby-2.5.0-r59376/libruby.so.2.5.0 (vm_call_cfunc+0xec) [0xffff940ae34] vm_insnhelper.c:1889\n*+
| */build\dir/build/\BUI\LD/ruby-2.5.0-r59376/libruby.so.2.5.0 (vm_call_method+0xc8) [0xffff940ac9e0] /include/ruby/ruby.h:1966\n*+
| */build\dir/build/\BUI\LD/ruby-2.5.0-r59376/libruby.so.2.5.0 (vm_exec_core+0x1490) [0xffff940a5ab8] insns.def:856\n*+
| */build\dir/build/\BUI\LD/ruby-2.5.0-r59376/libruby.so.2.5.0 (vm_exec+0x84) [0xffff940a97bc] vm_insnhelper.h:231\n*+
| */build\dir/build/\BUI\LD/ruby-2.5.0-r59376/libruby.so.2.5.0 (ruby_exec_internal+0xb4) [0xffff93f93f95] 3f993c4] eval.c:1244\n*+
| */build\dir/build/\BUI\LD/ruby-2.5.0-r59376/libruby.so.2.5.0 (ruby_exec_node+0x20) [0xffff93f97808] eval.c:308\n*+
| */build\dir/build/\BUI\LD/ruby-2.5.0-r59376/libruby.so.2.5.0 (ruby_run_node+0x24) [0xffff93f995 03/19/2020
After 6 patterns with 326 characters.

History

#1 - 09/01/2017 12:29 AM - naruse (Yui NARUSE)
- Status changed from Open to Feedback

I can't reproduce on ruby -v: ruby 2.5.0dev (2017-09-01 trunk 59707) [aarch64-linux].
see also http://rubyci.s3.amazonaws.com/scw-ad7f67/ruby-trunk/recent.html

#2 - 09/21/2017 10:32 AM - vo.x (Vit Ondruch)
naruse (Yui NARUSE) wrote:

I can't reproduce on ruby -v: ruby 2.5.0dev (2017-09-01 trunk 59707) [aarch64-linux].
see also http://rubyci.s3.amazonaws.com/scw-ad7f67/ruby-trunk/recent.html

You are using way older components then we do :/ I would blame glibc, binutils or something like that, since it seems the issue is in
rb_print_backtrace or even in backtrace(void **trace, int size) function.

I think that the test case, although it should segfault, segfaults on different place then expected. IOW it segfaults in segfault handler ...

Trying to reproduce this on my computer (x86_64) with following steps:

$ ruby -e 'puts "f" * 100' > test_ruby_test_bug7597
$ ruby --disable-gems -e "$0=ARGV[0]; Process.kill :SEGV, $$" test_ruby_test_bug7597

it actually takes down the bash instance it is running within. Is this expected?

#3 - 09/21/2017 03:34 PM - Eregon (Benoit Daloze)
vo.x (Vit Ondruch) wrote:

it actually takes down the bash instance it is running within. Is this expected?

Yes, Bash interprets the $$ inside the "":

```
    echo "$$"
    17490

    echo '$$'
    $$
```

#4 - 09/25/2017 08:15 AM - naruse (Yui NARUSE)
Eregon (Benoit Daloze) wrote:

vo.x (Vit Ondruch) wrote:

it actually takes down the bash instance it is running within. Is this expected?

Yes, Bash interprets the $$ inside the "":

```
    echo "$$"
    17490

    echo '$$'
    $$
```

Therefore use Process.pid instead.

#5 - 10/11/2017 03:46 PM - vo.x (Vit Ondruch)
Eregon (Benoit Daloze) wrote:

vo.x (Vit Ondruch) wrote:

it actually takes down the bash instance it is running within. Is this expected?

Yes, Bash interprets the $$ inside the "":

```
    echo "$$"
    17490

    echo '$$'
    $$
```

Oh my, how could I ... Thx :) 

Just FYI, I opened ticket with Fedora glibc maintainers, since it appears to be issue on some lower layer. I.e. it appears to fail in Kernel backtrace
function 1 for some reasons.

Interestingly, I was not able to reproduce outside of the test suite :)

#6 - 10/11/2017 03:47 PM - vo.x (Vit Ondruch)
vo.x (Vit Ondruch) wrote:
Just FYI, I opened ticket with Fedora glibc maintainers

Forgot to link the ticket: https://bugzilla.redhat.com/show_bug.cgi?id=1500863

#7 - 10/11/2017 03:50 PM - vo.x (Vit Ondruch)
BTW it seems that Ruby supports libunwind and we have libunwind available on Fedora. What would be the benefit of using it?

#8 - 10/12/2017 05:23 AM - vo.x (Vit Ondruch)
Can anybody help me we answer to glibc maintainers?

Florian Weimer 2017-10-11 18:00:03 CEST

Does Ruby call backtrace from a signal handler? It does on x86-64:

```
#0 __GL__backtrace (array=0x7f9c5cd12660, size=1024) at ../sysdeps/x86_64/backtrace.c:96
#1 0x00007f9c5ca27715 in rb_print_backtrace () from /lib64/libruby.so.2.4
#2 0x00007f9c5ca2794c in rb_vm_bugreport () from /lib64/libruby.so.2.4
#3 0x00007f9c5c900984 in rb_bug_context () from /lib64/libruby.so.2.4
#4 0x00007f9c5c9bce0e in sigsegv () from /lib64/libruby.so.2.4
#5 <signal handler called>
#6 0x00007f9c5ca16861 in vm_exec_core () from /lib64/libruby.so.2.4
#7 0x00007f9c5ca1b058 in vm_exec () from /lib64/libruby.so.2.4
#8 0x00007f9c5ca1b609 in eval_string_with_cref () from /lib64/libruby.so.2.4
#9 0x00007f9c5ca1c068 in rb_f_eval () from /lib64/libruby.so.2.4
```

That's not valid in its own right because backtrace can call malloc, and the SIGSEGV handler might be called from within malloc.

The backtrace issue we had on aarch64 has supposedly been fixed in rawhide:

https://sourceware.org/bugzilla/show_bug.cgi?id=21428

#9 - 10/12/2017 07:08 AM - normalperson (Eric Wong)

v.ondruch@tiscali.cz wrote:

Can anybody help me we answer to glibc maintainers [1]?

Florian Weimer 2017-10-11 18:00:03 CEST

Does Ruby call backtrace from a signal handler? It does on x86-64:

Yes, it calls backtrace() unfortunately. It uses special signal handlers for SIGILL, SIGSEGV, SIGBUS which sometimes screw up debugging.

When tracking down some bugs in the past; I've flipped the value of ruby_enable_coredump in signal.c to disable the nanny sighandlers and get a real core dump.

(there's a more official way which involves re-running ./configure and using RUBY_DEBUG env; but that's too slow for my hardware and I'd rather just recompile signal.o)

#10 - 12/14/2017 01:37 PM - vo.x (Vit Ondruch)

This is the glibc maintainer response:

```
[...]
```

Okay, this makes it more of a Ruby bug, unfortunately.

It would be nice if this was fixed in Ruby.

#11 - 11/29/2018 02:14 PM - vo.x (Vit Ondruch)

Any chance to get this fixed? It is nice that Ruby has such fancy SIGSEGV handler, but it does not work on all platforms and it makes the matters just worse.

#12 - 01/09/2020 03:01 PM - vo.x (Vit Ondruch)
Was there some change/improvement in this area? I am trying Ruby 2.7.0 and I have not seen this issue during several past build attempts. Therefore I wonder if this was fixed or I am just lucky, because this used to fail annoyingly often.