Ruby trunk - Bug #7084

RubyVM::InstructionSequence.compile("1+"*10000 + "1") causes SystemStackError or Segmentation Fault

09/29/2012 06:27 PM - mrkn (Kenta Murata)

Status:	Closed		
Priority:	Normal		
Assignee:	ko1 (Koichi Sasada)		
Target version:	2.0.0		
ruby -v:	ruby 2.0.0dev (2012-09-29 trunk 37053) [x86_64-darwin12.1.0]	**Backport:**	

Description

□□□□□□ ulimit -s □□□□□□ stack level too deep □ Segmentation Fault □□□□□□□□□□□□□□□□

```
$ .prefix/bin/ruby -v
ruby 2.0.0dev (2012-09-29 trunk 37053) [x86_64-darwin12.1.0]
$ ulimit -s
8192
$ .prefix/bin/ruby -e 'p RubyVM::InstructionSequence.compile("1+"*10000+"1")'
-e:1: stack level too deep (SystemStackError)
$ ulimit -s 32768
$ .prefix/bin/ruby -e 'p RubyVM::InstructionSequence.compile("1+"*10000+"1")'
-e:1: [BUG] Segmentation fault
ruby 2.0.0dev (2012-09-29 trunk 37053) [x86_64-darwin12.1.0]

-- Control frame information -----------------------------------------------
c:0003 p:---- s:0009 e:000008 CFUNC  :compile
c:0002 p:0028 s:0005 e:000bc8 EVAL   -e:1 [FINISH]
c:0001 p:0000 s:0002 e:0020e8 TOP    [FINISH]

-e:1:in <main>'
-e:1:incompile'

-- C level backtrace information -------------------------------------------

See Crash Report log file under ~/Library/Logs/CrashReporter or
/Library/Logs/CrashReporter, for the more detail of.

-- Other runtime information -----------------------------------------------
```

- Loaded script: -e
- Loaded features:

Associated revisions

Revision 54c17dc8 - 10/03/2012 06:33 AM - nobu (Nobuyoshi Nakada)

thread_pthread.c: init stack with ulimit

- thread_pthread.c (ruby_init_stack): use getrlimit() for the main thread on Mac OS X, since pthread_get_stack{addr,size}_np() and return the default value always, but not the ulimit value. [ruby-dev:46174] [Bug #7084]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@37072 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 37072 - 10/03/2012 06:33 AM - nobu (Nobuyoshi Nakada)

thread_pthread.c: init stack with ulimit

- thread_pthread.c (ruby_init_stack): use getrlimit() for the main thread on Mac OS X, since pthread_get_stack{addr,size}_np() and return the default value always, but not the ulimit value. [ruby-dev:46174] [Bug #7084]

Revision 37072 - 10/03/2012 06:33 AM - nobu (Nobuyoshi Nakada)

thread_pthread.c: init stack with ulimit

- thread_pthread.c (ruby_init_stack): use getrlimit() for the main thread on Mac OS X, since pthread_get_stack{addr,size}_np() and return the default value always, but not the ulimit value. [ruby-dev:46174] [Bug #7084]

Revision 37072 - 10/03/2012 06:33 AM - nobu (Nobuyoshi Nakada)

thread_pthread.c: init stack with ulimit

- thread_pthread.c (ruby_init_stack): use getrlimit() for the main thread on Mac OS X, since pthread_get_stack{addr,size}_np() and return the default value always, but not the ulimit value. [ruby-dev:46174] [Bug #7084]

Revision 37072 - 10/03/2012 06:33 AM - nobu (Nobuyoshi Nakada)

thread_pthread.c: init stack with ulimit

- thread_pthread.c (ruby_init_stack): use getrlimit() for the main thread on Mac OS X, since pthread_get_stack{addr,size}_np() and return the default value always, but not the ulimit value. [ruby-dev:46174] [Bug #7084]

Revision 37072 - 10/03/2012 06:33 AM - nobu (Nobuyoshi Nakada)

thread_pthread.c: init stack with ulimit

- thread_pthread.c (ruby_init_stack): use getrlimit() for the main thread on Mac OS X, since pthread_get_stack{addr,size}_np() and return the default value always, but not the ulimit value. [ruby-dev:46174] [Bug #7084]

Revision 37072 - 10/03/2012 06:33 AM - nobu (Nobuyoshi Nakada)

thread_pthread.c: init stack with ulimit

- thread_pthread.c (ruby_init_stack): use getrlimit() for the main thread on Mac OS X, since pthread_get_stack{addr,size}_np() and return the default value always, but not the ulimit value. [ruby-dev:46174] [Bug #7084]

Revision fab7e661 - 10/04/2012 02:43 AM - nobu (Nobuyoshi Nakada)

thread_pthread.c: precise stack size

- thread_pthread.c (ruby_init_stack): round stack limit to page size boundary to calculate stack size more precisely. [ruby-dev:46174] [Bug #7084]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@37080 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 37080 - 10/04/2012 02:43 AM - nobu (Nobuyoshi Nakada)

thread_pthread.c: precise stack size

- thread_pthread.c (ruby_init_stack): round stack limit to page size boundary to calculate stack size more precisely. [ruby-dev:46174] [Bug #7084]

Revision 37080 - 10/04/2012 02:43 AM - nobu (Nobuyoshi Nakada)

thread_pthread.c: precise stack size

- thread_pthread.c (ruby_init_stack): round stack limit to page size boundary to calculate stack size more precisely. [ruby-dev:46174] [Bug #7084]

Revision 37080 - 10/04/2012 02:43 AM - nobu (Nobuyoshi Nakada)

thread_pthread.c: precise stack size

- thread_pthread.c (ruby_init_stack): round stack limit to page size boundary to calculate stack size more precisely. [ruby-dev:46174] [Bug #7084]

Revision 37080 - 10/04/2012 02:43 AM - nobu (Nobuyoshi Nakada)

thread_pthread.c: precise stack size

- thread_pthread.c (ruby_init_stack): round stack limit to page size boundary to calculate stack size more precisely. [ruby-dev:46174] [Bug #7084]

Revision 37080 - 10/04/2012 02:43 AM - nobu (Nobuyoshi Nakada)

thread_pthread.c: precise stack size

- thread_pthread.c (ruby_init_stack): round stack limit to page size boundary to calculate stack size more precisely. [ruby-dev:46174] [Bug #7084]

Revision 37080 - 10/04/2012 02:43 AM - nobu (Nobuyoshi Nakada)

thread_pthread.c: precise stack size

- thread_pthread.c (ruby_init_stack): round stack limit to page size boundary to calculate stack size more precisely. [ruby-dev:46174] [Bug #7084]

History

#1 - 09/29/2012 06:29 PM - mrkn (Kenta Murata)

gdb で採取したバックとレースを gist に上げました。ulimit -s 8192 の場合です。参考までに。
https://gist.github.com/3803540

#2 - 09/29/2012 07:53 PM - ko1 (Koichi Sasada)

(2012/09/29 18:27), mrkn (Kenta Murata) wrote:

> 以下のように ulimit -s の値に応じて stack level too deep と Segmentation Fault のどちらかが発生するようです。

□ええと，どうしたもんですかね．実装の都合でスタックオーバーフローするも
のは他にもありそうなので，これだけ頑張るべきか，よくわからないんですが....

□これだけ取り出した理由はなんでしょうか．こういう操作ってよくやるんです
かね．

--
// SASADA Koichi at atdot dot net

#3 - 09/29/2012 11:11 PM - mrkn (Kenta Murata)

"1+"*10000 + "1" は□YAPC::Asia の LT で見かけたネタです。

ulimit -s の値に依存して SEGV してしまう場合があるので、それだけでも直したいです。

#4 - 09/30/2012 08:23 PM - ko1 (Koichi Sasada)

(2012/09/29 23:11), mrkn (Kenta Murata) wrote:

> ulimit -s の値に依存して SEGV してしまう場合があるので、それだけでも直したいです。

□ulimit -s の値に依存せず SEGV するのはもっと嫌な気がしますが，それはと
もかく，直るのなら大変結構なことだと思います．よろしくお願いします．

--
// SASADA Koichi at atdot dot net

#5 - 10/03/2012 03:33 PM - nobu (Nobuyoshi Nakada)

- Status changed from Open to Closed

- % Done changed from 0 to 100

This issue was solved with changeset r37072.
Kenta, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

---

thread_pthread.c: init stack with ulimit

- thread_pthread.c (ruby_init_stack): use getrlimit() for the main thread on Mac OS X, since pthread_get_stack{addr,size}_np() and return the default value always, but not the ulimit value. [ruby-dev:46174] [Bug #7084]

#6 - 10/03/2012 06:38 PM - ko1 (Koichi Sasada)

これって，r37072 で直ったんでしょうか．

#7 - 10/04/2012 02:22 AM - mrkn (Kenta Murata)

- Status changed from Closed to Open

まだ直ってません。ulimit -s 32768 の場合はきちんと SystemStackError で終了しましたが、
ulimit -s 4096 の場合に Segmentation Fault が発生しています。

さっき追加で調査した結果を以下に述べます。
https://gist.github.com/3828296
この gist に貼った 1.patch を当てて問題をコードを実行すると、

1.logとして貼付けたようにiseq_compile_eachのローカル変数typeのアドレスがずらずら表示されます。

前後に隣り合うアドレスの差分を計算すると、再帰呼び出し1段でtypeのアドレスが6928も変化していてページサイズより大きいため、
ruby_stack_overflowed_pでスタックオーバーフローの判定に失敗しているのかなぁと推測しました。

そこで、iseq_compile_eachの中のswitch文を複数の関数に分離して実行してみました。
https://gist.github.com/3828416

iseq_compile_eachが使うスタックのサイズは560まで減りましたが、依然としてスタックオーバーフローの判定に失敗してます。

ここまでやって力尽きたので、また日を改めて調査します。

#8 - 10/04/2012 11:43 AM - nobu (Nobuyoshi Nakada)

- Status changed from Open to Closed

This issue was solved with changeset r37080.
Kenta, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

---

thread_pthread.c: precise stack size

- thread_pthread.c (ruby_init_stack): round stack limit to page size boundary to calculate stack size more precisely. [ruby-dev:46174] [Bug #7084]