Ruby master - Bug #15430

test_fork_while_parent_locked is failing status on Ruby CI

12/18/2018 01:11 AM - hsbt (Hiroshi SHIBATA)

<table>
<thead>
<tr>
<th>Status:</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td>normalperson</td>
</tr>
<tr>
<td>Target version:</td>
<td>ruby-v:</td>
</tr>
<tr>
<td>Backport:</td>
<td>2.4: UNKNOWN, 2.5: UNKNOWN</td>
</tr>
</tbody>
</table>

Description
After r66230, TestThread#test_fork_while_parent_locked is failing status on Ruby CI.

See

Associated revisions

Revision 1df9c2bc - 12/18/2018 09:21 AM - normal

thread_sync.c (mutex_ptr): only reinitialize waitqueue at fork

Mutexes need to remain locked after forking.

This fixes "[BUG] invalid keeping_mutexes: Attempt to unlock a mutex which is locked by another thread" and should fix test_fork_while_parent_locked failures in CI

[ruby-core:90581] [Bug #15424]
[ruby-core:90595] [Bug #15430]

Fixes: r66230 ("handle mutexes held by parent threads in children")
git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66438 b2dd03c8-39d4-4d8f-98f1-823fe69b080e

Revision 66438 - 12/18/2018 09:21 AM - normal

thread_sync.c (mutex_ptr): only reinitialize waitqueue at fork

Mutexes need to remain locked after forking.

This fixes "[BUG] invalid keeping_mutexes: Attempt to unlock a mutex which is locked by another thread" and should fix test_fork_while_parent_locked failures in CI

[ruby-core:90581] [Bug #15424]
[ruby-core:90595] [Bug #15430]

Fixes: r66230 ("handle mutexes held by parent threads in children")

Revision 66438 - 12/18/2018 09:21 AM - normal

thread_sync.c (mutex_ptr): only reinitialize waitqueue at fork

Mutexes need to remain locked after forking.

This fixes "[BUG] invalid keeping_mutexes: Attempt to unlock a mutex which is locked by another thread" and should fix test_fork_while_parent_locked failures in CI

[ruby-core:90581] [Bug #15424]
[ruby-core:90595] [Bug #15430]

Fixes: r66230 ("handle mutexes held by parent threads in children")

09/19/2021
thread_sync.c (rb_mutex_cleanup_keeping_mutexes): update fork_gen

... when clearing waitq. Otherwise, we risk redundantly clearing valid waiters in future calls to mutex_ptr.

Note: I am not sure if this fixes [Bug #15430], and even if it did, fork_gen is a belt-and-suspenders redundancy for [Bug #15383] which wastes one word for every Mutex object.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66477 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 66477 - 12/21/2018 12:32 PM - normal
thread_sync.c (rb_mutex_cleanup_keeping_mutexes): update fork_gen

... when clearing waitq. Otherwise, we risk redundantly clearing valid waiters in future calls to mutex_ptr.

Note: I am not sure if this fixes [Bug #15430], and even if it did, fork_gen is a belt-and-suspenders redundancy for [Bug #15383] which wastes one word for every Mutex object.

Revision 66477 - 12/21/2018 12:32 PM - normal
thread_sync.c (rb_mutex_cleanup_keeping_mutexes): update fork_gen

... when clearing waitq. Otherwise, we risk redundantly clearing valid waiters in future calls to mutex_ptr.

Note: I am not sure if this fixes [Bug #15430], and even if it did, fork_gen is a belt-and-suspenders redundancy for [Bug #15383] which wastes one word for every Mutex object.

Revision 691160f - 12/21/2018 09:44 PM - normal

test/ruby/test_thread.rb: add diagnosis code for [Bug #15430]

I can't find stderr in the test-all output of the CI machine, so maybe the assertion will show what's going on.


git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66487 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 66487 - 12/21/2018 09:44 PM - normal

test/ruby/test_thread.rb: add diagnosis code for [Bug #15430]

I can't find stderr in the test-all output of the CI machine, so maybe the assertion will show what's going on.


Revision 66487 - 12/21/2018 09:44 PM - normal

test/ruby/test_thread.rb: add diagnosis code for [Bug #15430]

I can't find stderr in the test-all output of the CI machine, so maybe the assertion will show what's going on.


Revision 0fd35f51 - 12/22/2018 01:41 AM - normal

thread_sync.c (rb_mutex_t): eliminate fork_gen

The true bug fork_gen was hiding was rb_mutex_abandon_locking_mutex failing to unconditionally clear the waitq of mutexes it was waiting on. So we fix rb_mutex_abandon_locking_mutex, instead, and eliminate rb_mutex_cleanup_keeping_mutexes.

This commit was tested heavily on a single-core Pentium-M which was my most reliable reproducer of the "crash.rb" script from [Bug #15383]
Note: [Bug #15430] turned out to be an entirely different problem: RLIMIT_NPROC limit was hit on the CI VMs.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66489 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 66489 - 12/22/2018 01:41 AM - normalperson (Eric Wong)

thread_sync.c (rb_mutex_t): eliminate fork_gen

The true bug fork_gen was hiding was rb_mutex_abandon_locking_mutex failing to unconditionally clear the waitq of mutexes it was waiting on. So we fix rb_mutex_abandon_locking_mutex, instead, and eliminate rb_mutex_cleanup_keeping_mutexes.

This commit was tested heavily on a single-core Pentium-M which was my most reliable reproducer of the "crash.rb" script from [Bug #15383]

Note: [Bug #15430] turned out to be an entirely different problem: RLIMIT_NPROC limit was hit on the CI VMs.

Revision 66489 - 12/22/2018 01:41 AM - normal

thread_sync.c (rb_mutex_t): eliminate fork_gen

The true bug fork_gen was hiding was rb_mutex_abandon_locking_mutex failing to unconditionally clear the waitq of mutexes it was waiting on. So we fix rb_mutex_abandon_locking_mutex, instead, and eliminate rb_mutex_cleanup_keeping_mutexes.

This commit was tested heavily on a single-core Pentium-M which was my most reliable reproducer of the "crash.rb" script from [Bug #15383]

Note: [Bug #15430] turned out to be an entirely different problem: RLIMIT_NPROC limit was hit on the CI VMs.

Revision db68ccc2 - 12/22/2018 08:08 AM - normal

test/ruby/test_thread.rb (test_fork_while_parent_locked): reduce threads

Reduce thread counts unconditionally for some CI systems with low limits.. And Solaris apparently lacks RLIMIT_NPROC, so we can't detect resource limits and scale the test appropriately.

[ruby-core:90670] [Bug #15430]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66494 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 66494 - 12/22/2018 08:08 AM - normalperson (Eric Wong)

test/ruby/test_thread.rb (test_fork_while_parent_locked): reduce threads

Reduce thread counts unconditionally for some CI systems with low limits.. And Solaris apparently lacks RLIMIT_NPROC, so we can't detect resource limits and scale the test appropriately.

[ruby-core:90670] [Bug #15430]

Revision 66494 - 12/22/2018 08:08 AM - normal

test/ruby/test_thread.rb (test_fork_while_parent_locked): reduce threads

Reduce thread counts unconditionally for some CI systems with low limits.. And Solaris apparently lacks RLIMIT_NPROC, so we can't detect resource limits and scale the test appropriately.

[ruby-core:90670] [Bug #15430]
Revision 37ba27b - 12/23/2018 08:42 AM - normal

test/ruby/test_thread.rb (test_fork_while_parent_locked): rewrite to avoid OOM
Instead of using a torture test, trigger the condition for the old segfault in [Bug #15383] exactly.
[ruby-core:90676] [Bug #15430]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66508 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 66508 - 12/23/2018 08:42 AM - normalperson (Eric Wong)
test/ruby/test_thread.rb (test_fork_while_parent_locked): rewrite to avoid OOM
Instead of using a torture test, trigger the condition for the old segfault in [Bug #15383] exactly.
[ruby-core:90676] [Bug #15430]

Revision 66508 - 12/23/2018 08:42 AM - normal

test/ruby/test_thread.rb (test_fork_while_parent_locked): rewrite to avoid OOM
Instead of using a torture test, trigger the condition for the old segfault in [Bug #15383] exactly.
[ruby-core:90676] [Bug #15430]

History

#1 - 12/18/2018 04:32 AM - normalperson (Eric Wong)
Thanks. Maybe related to [Bug #15430]. Will investigate in coming days.

#2 - 12/18/2018 09:21 AM - normalperson (Eric Wong)
- Status changed from Assigned to Closed

Applied in changeset trunk|r66438.

thread_sync.c (mutex_ptr): only reinitalize waitqueue at fork

Mutexes need to remain locked after forking.
This fixes "[BUG] invalid keeping_mutexes: Attempt to unlock a mutex which is locked by another thread" and should fix test_fork_while_parent_locked failures in CI
[ruby-core:90581] [Bug #15424]
[ruby-core:90595] [Bug #15430]

Fixes: r66230 ("handle mutexes held by parent threads in children")

#3 - 12/18/2018 09:52 AM - hsbt (Hiroshi SHIBATA)
r66438 should fix it. Ugh, I clearly shouldn't be allowed near a keyboard anymore :<

#4 - 12/19/2018 02:03 AM - normalperson (Eric Wong)
- Status changed from Closed to Assigned

normalperson (Eric Wong)


#5 - 12/20/2018 08:04 AM - normalperson (Eric Wong)


09/19/2021
I am now thinking the r66230 for [Bug #15383] is unnecessary and [Bug #15383] is a bug somewhere else.

But I can't reproduce it atm...

#6 - 12/20/2018 09:22 AM - normalperson (Eric Wong)

But I can't reproduce it atm...

OK, I need a break and get some sleep (but I realize 2.6 release is is very soon).

I can reproduce [Bug #15383] with "rb_mutex_t.fork_gen" field removed, but fork_gen shouldn't be necessary in mutex because of rb_mutex_abandon_* functions and rb_mutex_cleanup_keeping_mutexes already call list_head_init(waitq).

So there is a bug in one of the rb_mutex_[abandon,cleanup]* functions which is hidden by the addition of rb_mutex_t.fork_gen.

#7 - 12/21/2018 12:32 PM - normalperson (Eric Wong)

- Status changed from Assigned to Closed

Applied in changeset trunk|r66477.

thread_sync.c (rb_mutex_cleanup_keeping_mutexes): update fork_gen

... when clearing waitq. Otherwise, we risk redundantly clearing valid waiters in future calls to mutex_ptr.

Note: I am not sure if this fixes [Bug #15430], and even if it did, fork_gen is a belt-and-suspenders redundancy for [Bug #15383] which wastes one word for every Mutex object.

#8 - 12/21/2018 10:42 PM - normalperson (Eric Wong)

Fwiw, I think the bug is in rb_mutex_abandon_locking_mutex, it needs to clear the waitq unconditionally; but not touch mutex->th or mutex->next_mutex:

https://80x24.org/spew/20181221223422.9219-1-e@80x24.org/raw

And as a result, rb_mutex_cleanup_keeping_mutexes is redundant.

#9 - 12/22/2018 07:19 AM - k0kubun (Takashi Kokubun)

- Status changed from Closed to Assigned

As of r66490, the test on some CIs is still failing for different reasons.

Expected "if" to be empty.


ArgumentError: invalid resource name: NPROC


Could you take a look at it again?
test/ruby/test_thread.rb (test_fork_while_parent_locked): reduce threads

Reduce thread counts unconditionally for some CI systems with low limits. And Solaris apparently lacks RLIMIT_NPROC, so we can't detect resource limits and scale the test appropriately.

[ruby-core:90670] [Bug #15430]

#11 - 12/22/2018 08:22 AM - normalperson (Eric Wong)
takashikbn@gmail.com wrote:

As of r66490, the test on some CIs is still failing for different reasons.

Expected "#" to be empty.

OK, I lowered thread counts in r66494.

Btw, can we get dmesg or kernel logs from those CI machines to confirm it was OOM killer? I guess RLIMIT_NPROC was a bad check anyways; and OOM killer is a little unpredictable...

ArgumentError: invalid resource name: NPROC

Interesting; I haven't touched Solaris in ages but I never thought it lacked RLIMIT_NPROC. So I unconditionally removed the check.

#12 - 12/23/2018 12:57 AM - k0kubun (Takashi Kokubun)

As of r66497 or r66498, some Linux environments are getting:

1) Failure:
TestThread#test_fork_while_parent_locked [/home/hsbt/chkbuild/tmp/build/20181222T213003Z/ruby/test/ruby/test_thread.rb:1278]:
lower `nr' if SIGKILL because of RLIMIT_NPROC limit.
Expected "#<Process::Status: pid 17656 SIGKILL (signal 9)>" to be empty.


So reopening this. If it's just a bad check, please consider eliminating the check or skipping the test itself to avoid false-positive alarms.

Btw, can we get dmesg or kernel logs from those CI machines to confirm it was OOM killer?

I don't have the access to most of RubyCI servers. Please contact hsbt (Hiroshi SHIBATA), naruse (Yui NARUSE), or possibly akr (Akira Tanaka) about those servers.

#13 - 12/23/2018 08:42 AM - normalperson (Eric Wong)

- Status changed from Assigned to Closed

Applied in changeset trunk\r66508.

TestThread#test_fork_while_parent_locked: rewrite to avoid OOM

Instead of using a torture test, trigger the condition for the old segfault in [Bug #15383] exactly.
So reopening this. If it's just a bad check, please consider eliminating the check or skipping the test itself to avoid false-positive alarms.

OK, I rewrote the test to surgically trigger the condition r66508. Feel free to delete the test if it still fails; I don't think I'll be around much more.

It seems to be all green now. Thank you for your help.