Hi,

I'm reporting a reliable crash of the ruby interpreter on contested mutexses that are accessed in child processes.

I currently think that this happens as the child processes main thread, may be waiting for a parent process sibling thread that was holding the mutex at the time of the fork. After the fork is done, all sibling threads are dead, and the mutex detects the attempt to wait for a dead thread, bailing out.

This is similar, but not identical to the case here: https://bugs.ruby-lang.org/issues/14578

Here is a gist with some more test results on various platforms: https://gist.github.com/mbj/e6795ee5e0583c5541ee250e9942279a

I'm fine to get my hands dirty, but would need some pointers if my above conclusion points to the right direction.

Best,
Markus

Related issues:
Related to Ruby master - Bug #14634: Queue#push seems to crash after fork

Associated revisions
Revision 818f1c65 - 12/05/2018 06:58 PM - normal
thread_sync.c (mutex_ptr): handle mutexes held by parent threads in children

Mutexes may be held by threads which only exist in the parent process, so their waitqueues may be populated with references to other dead threads. We must reset them at fork.

I am a moron for introducing this bug :<

[ruby-core:90312] [Bug #15383]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66230 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 66230 - 12/05/2018 06:58 PM - normalperson (Eric Wong)
thread_sync.c (mutex_ptr): handle mutexes held by parent threads in children

Mutexes may be held by threads which only exist in the parent process, so their waitqueues may be populated with references to other dead threads. We must reset them at fork.

I am a moron for introducing this bug :<

[ruby-core:90312] [Bug #15383]

Revision 66230 - 12/05/2018 06:58 PM - normal
thread_sync.c (mutex_ptr): handle mutexes held by parent threads in children

Mutexes may be held by threads which only exist in the parent process, so their waitqueues may be populated with references to other dead threads. We must reset them at fork.
I am a moron for introducing this bug :<

[ruby-core:90312] [Bug #15383]

Revision fa5601e7 - 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.

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
person (Eric Wong)
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
person (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]

[Bug #14578] [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 - normal
person (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]

[Bug #14578] [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]
[Bug #14578] [Bug #15383]

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

Revision 37bba27b - 12/23/2018 08:42 AM - normal

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]
[Bug #14578] [Bug #15383]

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

Revision 66508 - 12/23/2018 08:42 AM - normalperson (Eric Wong)

thread_sync.c (mutex_ptr): handle mutexes held by parent threads in children

Mutexes may be held by threads which only exist in the parent
process, so their waitqueues may be populated with references
to other dead threads. We must reset them at fork.

I am a moron for introducing this bug :<

[Bug #15383]

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

History

#1 - 12/05/2018 06:57 PM - normalperson (Eric Wong)
- Backport changed from 2.4: UNKNOWN, 2.5: UNKNOWN to 2.4: UNKNOWN, 2.5: REQUIRED

#2 - 12/05/2018 06:58 PM - normalperson (Eric Wong)
- Status changed from Open to Closed

Applied in changeset trunkr66230.

thread_sync.c (mutex_ptr): handle mutexes held by parent threads in children

Mutexes may be held by threads which only exist in the parent
process, so their waitqueues may be populated with references
to other dead threads. We must reset them at fork.

I am a moron for introducing this bug :<

[ruby-core:90312] [Bug #15383]

#3 - 12/05/2018 07:03 PM - normalperson (Eric Wong)
Thanks, it affects trunk; just more difficult to reproduce because of thread cache.

I'm a moron for not noticing this when I fixed other bugs :<

r66230 should fix it in trunk and should be backported (but r66229 is independently broken and I just reverted it for now)

#4 - 12/05/2018 07:36 PM - mbjs (Markus Schirp)

#5 - 12/05/2018 07:37 PM - mbjs (Markus Schirp)
Thanks for the quick fix. Also for marking the fix to be backported.

Just curious, is there an associated CI build for these changes?

#6 - 12/05/2018 08:22 PM - normalperson (Eric Wong)
mbj@schirp-dso.com wrote:

Just curious, is there an associated CI build for these changes?

I check https://rubyci.org/ and http://ci.rvm.jp/ (and get automated mails from the latter).

There's also TravisCI; but I don't use JavaScript; so I rely on others giving me URLs to the raw logs.

#7 - 12/06/2018 12:10 AM - nagachika (Tomoyuki Chikanaga)
- Related to Bug #14634: Queue#push seems to crash after fork added

#8 - 12/20/2018 08:42 AM - normalperson (Eric Wong)

https://bugs.ruby-lang.org/issues/15383

r66230 should fix it in trunk and should be backported

No, actually. r66230 hides an existing problem in the fix for https://bugs.ruby-lang.org/issues/14578

... Still working on this and my head hurts :<

Files

<table>
<thead>
<tr>
<th>File</th>
<th>Size</th>
<th>Date</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>output.txt</td>
<td>21.3 KB</td>
<td>12/05/2018</td>
<td>mbjs (Markus Schirp)</td>
</tr>
<tr>
<td>crash.rb</td>
<td>200 Bytes</td>
<td>12/05/2018</td>
<td>mbjs (Markus Schirp)</td>
</tr>
</tbody>
</table>