Ruby - Bug #10443

Forking with contended mutex held can lead to deadlock in child process upon unlock

10/28/2014 03:42 PM - benweint (Ben Weintraub)

Status:	Closed		
Priority:	Normal		
Assignee:	kosaki (Motohiro KOSAKI)		
Target version:			
ruby -v:	ruby 2.1.3p242 (2014-09-19 revision 47630) [x86_64-darwin13.0]	**Backport:**	2.0.0: UNKNOWN, 2.1: UNKNOWN

Description

If a Ruby thread calls Process.fork while holding a Mutex (for example, within a Mutex#synchronize block) that is also concurrently being contended for by a background thread, the thread in the child process will occasionally be unable to unlock the mutex it was holding at the time of the fork, and will hang under rb_mutex_unlock_th when attempting to acquire mutex->lock.

I've been able to reproduce this on Ruby 2.1.1 - 2.1.3 and 2.2.0-preview1 (haven't tried elsewhere yet).

The attached test case demonstrates the issue, although it can take up to 20 minutes to hit a reproducing case. The test case will print one '.' each time it forks. Once it stops printing dots, it has hit this bug (the parent process is stuck in a call to Process.wait, and the child is stuck in rb_mutex_unlock_th).

The test case consists of a global lock that is contended for by 10 background threads, in addition to the main thread, which acquires it, forks, and then releases it.

History

#1 - 10/28/2014 07:03 PM - benweint (Ben Weintraub)

- File rb-mutex-unlock-fork-test.rb added

The original test case was not actually minimal (there's no need to attempt to re-acquire the lock in the forked child process in order to demonstrate the issue), so I'm attaching an updated version.

#2 - 10/28/2014 08:32 PM - normalperson (Eric Wong)

Thanks for the test case, I can reproduce it easily.
Hopefully I can fix it soon.

#3 - 10/28/2014 09:52 PM - normalperson (Eric Wong)

- File 0001-thread.c-reinitialize-keeping-mutexes-on-fork.patch added

- Category set to core

- Assignee set to kosaki (Motohiro KOSAKI)

- Target version set to 2.2.0

For now, I think leaking some lock/cond resources at fork is the easiest
option. Leaking is better than deadlocking.

kosaki: thoughts on my patch? Thanks.

#4 - 02/22/2015 05:42 AM - nobu (Nobuyoshi Nakada)

- Description updated

Seems linux specific issue.
I could reproduce it in a few iterations on Ubuntu 14 x86_64, but iterated successfully 1000 times on OSX.

So I think it would be better to separate the code from rb_thread_atfork() as a function.

```
diff --git a/thread.c b/thread.c
index 6fdacec..0fa2cbb 100644
--- a/thread.c
+++ b/thread.c
@@ -3862,15 +3862,12 @@ terminate_atfork_i(rb_thread_t *th, const rb_thread_t *current_th)
```

```
     }
 }

-void
-rb_thread_atfork(void)
+#ifdef __linux__
+static void
+thread_destroy_keeping_mutexes(rb_thread_t *th)
 {
-    rb_thread_t *th = GET_THREAD();
     size_t n = 0;
     rb_mutex_t *mutex;
-    rb_thread_atfork_internal(terminate_atfork_i);
-
-    th->join_list = NULL;

     /* we preserve mutex state across fork, but ensure we do not deadlock */
     mutex = th->keeping_mutexes;
@@ -3890,6 +3887,20 @@ rb_thread_atfork(void)
     if (n) {
  rb_warn("%"PRIuSIZE" Mutex resource(s) leaked on fork", n);
     }
+}
+#else
+#define thread_destroy_keeping_mutexes(th) /* do nothing */
+#endif
+
+void
+rb_thread_atfork(void)
+{
+    rb_thread_t *th = GET_THREAD();
+
+    rb_thread_atfork_internal(terminate_atfork_i);
+
+    th->join_list = NULL;
+    thread_destroy_keeping_mutexes(th);

     /* We don't want reproduce CVE-2003-0900. */
     rb_reset_random_seed();
```

#5 - 02/22/2015 06:56 AM - nobu (Nobuyoshi Nakada)

Maybe, thread_destroy_keeping_mutexes() should be in thread_pthread.c.

#6 - 01/05/2018 09:01 PM - naruse (Yui NARUSE)

- Target version deleted (2.2.0)

#7 - 07/22/2019 10:44 PM - jeremyevans0 (Jeremy Evans)

- Status changed from Open to Closed

I think this problem is fixed. The example given works on Linux and OpenBSD. If anyone is still having the problem with the master branch, please post back here.

Files

rb-mutex-unlock-fork-test.rb	372 Bytes	10/28/2014	benweint (Ben Weintraub)
rb-mutex-unlock-fork-test.rb	339 Bytes	10/28/2014	benweint (Ben Weintraub)
0001-thread.c-reinitialize-keeping-mutexes-on-fork.patch	3.06 KB	10/28/2014	normalperson (Eric Wong)