Does the thread id change after forking a new process?

Asked 6 years, 9 months ago Modified 1 year, 3 months ago Viewed 2k times

2

I know that this fork creates a new process, but what about the thread that was running prior to calling fork, does it also change (because now it is part of a new process "child process" which should have new threads ?)

Compiling and running the following C test confirms that the thread id remains the same:

```
pthread_t threadId1, threadId2;

threadId1 = pthread_self();
if (fork() == 0)
{
  threadId2 = pthread_self();
  if (pthread_equal(threadId1,threadId2)) // edited
  {
    printf("we are in the same thread \n");
  }
 else
  {
   printf("we are on different threads \n");
  }
```

Could someone explain to me why the thread is shared among the parent and child process ?

c multithreading pthreads fork

Share Improve this question Follow

edited Jan 20, 2017 at 9:24

asked Jan 20, 2017 at 9:10

Bionix1441

2,145 ● 1 ● 30 ● 65

4 Thread ids are *opaque* data, you should not use direct comparison of them. Instead use `pthread_equal`. – Some programmer dude Jan 20, 2017 at 9:13

1 Use gettid() rather than the opaque pthread_t structure. – Claudio Jan 20, 2017 at 9:15

Yes, I just compare and print the result of the comparison of thread id before forking and in the child process. – Bionix1441 Jan 20, 2017 at 9:16

1 I'm not sure, but maybe threads are tied up to a parent process and their ids are unique only within it. Therefore you can get the same ids. – Long Smith Jan 20, 2017 at 9:17

2 Moreover POSIX standard claims that comparing threads with '==' is not appropriate way. You have to use pthread_equal. – Long Smith Jan 20, 2017 at 9:20

2 Answers

Sorted by: Highest score (default)

4

If you read the `pthread_self` manual page you will see that

> Thread IDs are guaranteed to be unique only **within a process**.

(Emphasis mine)

That of course means that two very different processes may have threads with the same id.

If you for some reason want to get the unique kernel id of the thread, use `gettid` instead.

Share Improve this answer Follow

edited Jun 30, 2022 at 13:02

answered Jan 20, 2017 at 9:15

Some programmer dude

401k ● 35 ● 404 ● 623

I have missed that Thank you. – Bionix1441 Jan 20, 2017 at 9:18

Is there a 't' missing because the link is to gettid? – SoulfreezerXP Jun 30, 2022 at 12:57

@SoulfreezerXP Yes that was a typo. – Some programmer dude Jun 30, 2022 at 13:02

3

From the man pages of pthread_self

```
Thread IDs are guaranteed to be unique only within a process.  A
thread ID may be reused after a terminated thread has been joined, or
a detached thread has terminated.

The thread ID returned by pthread_self() is not the same thing as the
kernel thread ID returned by a call to gettid(2).
```

Since fork effectively duplicates a process, including its handles to kernel objects, the result therefore is not unexpected. (The kernel uses both handle value and pid when doing object lookup)

Share Improve this answer Follow

edited Jan 20, 2017 at 11:21

answered Jan 20, 2017 at 9:21

doron

28k ● 12 ● 65 ● 104