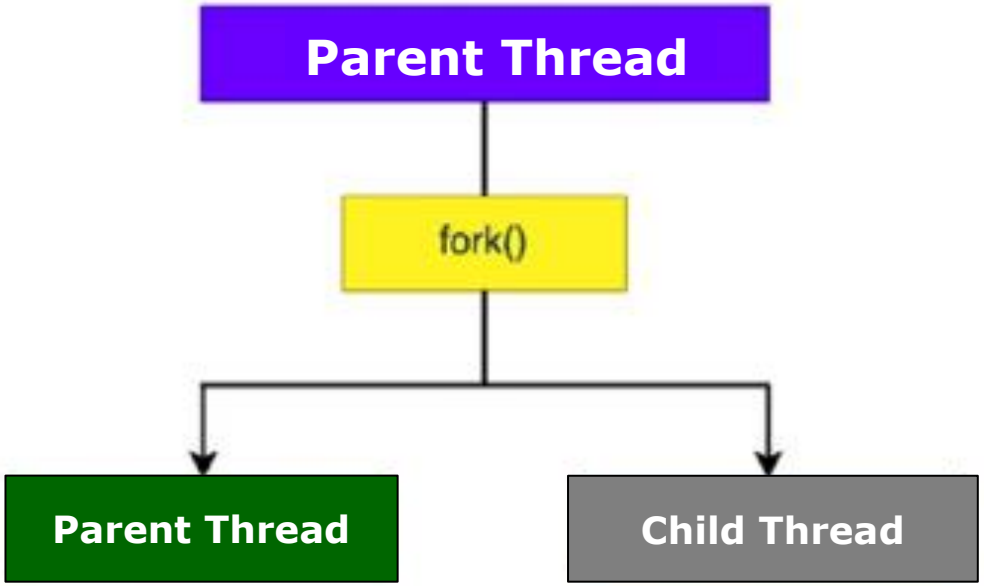
Fork() and Exec() system calls

fork() system call

Creating a thread is done with a **fork()** system call.

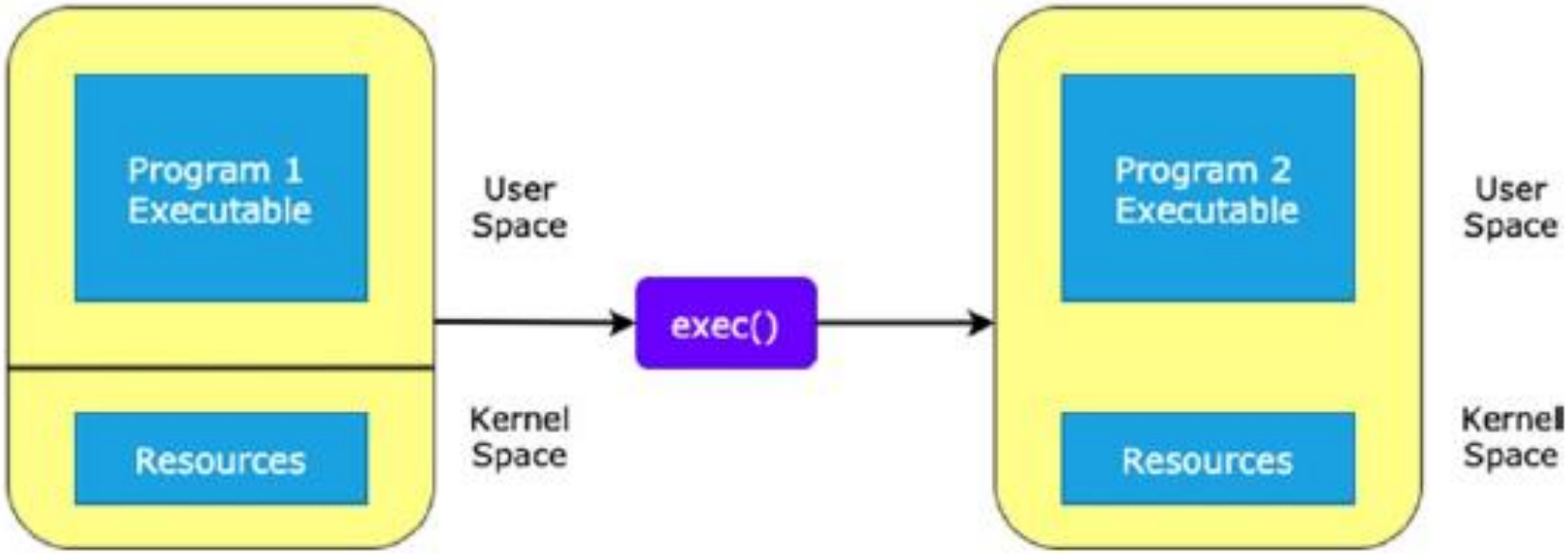
- ✓ A newly created thread is called a **child thread**, and the thread that is initiated to create the new thread is considered a **parent thread**.



exec() system call

*The **exec()** system call family replaces the currently running thread with a new thread.*

- ✓ The original thread identifier remains the same, and all the internal details, such as stack, data, and instructions.
- ✓ The new thread replaces the executables.



Semantics of fork() and exec()

Does **fork()** duplicate only the calling thread or all threads?

How should we implement fork?

- logical thing to do is:

  1) If **exec()** will be called after **fork()**, there is no need to duplicate the threads. They will be replaced anyway.

  2) If **exec()** will not be called, then it is logical to duplicate the threads as well; so that the child will have as many threads as the parent has.

- So, we may implement two system calls: like fork1 and fork2! *(**you can try this in the lab, later**)*