死锁：一种情形，此时执行程序中两个或多个线程发生永久堵塞（等待），每个线程都在等待被
其他线程占用并堵塞了的资源。例如，如果线程A锁住了记录1并等待记录2，而线程B锁住了记录2并等待记录1，这样两个线程就发生了死锁现象。

gdb调试死锁的方法：
gdb
attach pid
thread apply all bt

找到_lll_lock_wait 锁等待的地方。
然后查找该锁被哪个线程锁住了。
例如：

查看哪个线程拥有互斥体
(gdb) print AccountA_mutex
$1 = {__m_reserved = 2, __m_count = 0, __m_owner = 0x2527,
__m_kind = 0, __m_lock
= {__status = 1, __spinlock = 0}}
(gdb) print 0x2527
$2 = 9511
(gdb) print AccountB_mutex
$3 = {__m_reserved = 2, __m_count = 0, __m_owner = 0x2529,
__m_kind = 0, __m_lock = {__status = 1, __spinlock = 0}}
(gdb) print 0x2529
$4 = 9513
(gdb)
从上面的命令中，我们可以看出AccontA_mutex是被线程 5（LWP 9511）加锁（拥有）的，而AccontB_mutex是被线程 3（LWP 9513）加锁（拥有）的。

找出死锁的地方，对应检查代码就可以了。死锁大多是对锁的使用发生交叉所致的，解决死锁的方法常有：

有序资源分配法

银行算法

说明：AccountA_mutex 是锁变量，

$1 = {__m_reserved = 2, __m_count = 0, __m_owner = 0x2527,

__m_kind = 0, __m_lock

= {__status = 1, __spinlock = 0}}

__m_owner = 0x2527 是当前 锁的ower 线程 pid ，通过该 pid 可以找到对应的线程，查看代码，