**LwIP内核BUG测试说明**

采用阻塞模式时，连续两次与服务器建立、关闭连接时，在运行一段时候，发现“任务阻塞”，而系统能ping成功，究其原因，是任务阻塞在以下语句上：

connect(n->m\_nSockFd, (struct sockaddr\*)&address, sizeof(struct sockaddr\_in))

跟踪语句，采用“日志记录”定位，最后确定语句阻塞在：

lwip\_connect----->netconn\_connect------>lwip\_netconn\_do\_connect以下语句：

UNLOCK\_TCPIP\_CORE();

sys\_arch\_sem\_wait(LWIP\_API\_MSG\_SEM(msg), 0);

LOCK\_TCPIP\_CORE();

也就是等待信号量：conn.op\_completed\_sem，以期待动作执行完毕后，任务释放该信号量。

该信号量在lwip\_netconn\_do\_connected、tcp\_err等函数中释放post：

if (was\_blocking) {

sys\_sem\_signal(op\_completed\_sem);

}

**怀疑1：**是否移植函数sys\_sem\_signal释放不成功，在sys\_sem\_signal函数加入判断返回错误码，如果发生错误，就重复10次post，每次间隔10ms，在出现错误的情况下，同时记录日志，观察长时间，任务阻塞仍然出现，但是并没有记录到post信号量失败。因此该原因可以排除。

**怀疑2：**修改：

void sys\_sem\_signal(sys\_sem\_t \*sem)

{

OS\_ERR err;

OSSemPost(sem,OS\_OPT\_POST\_ALL,&err);//发送信号量

}

以前是OS\_OPT\_POST\_1，只对最高任务的发送信号量，而现在修改为OS\_OPT\_POST\_ALL对所有的任务都发送信号量。

运行一段时间后，结果仍然会“任务阻塞”。

**怀疑3：**由于LwIP的配置文件lwipopt.h中配置采用内存堆代替内存池分配，采用Heap代替内存堆分配，因此怀疑是内存原因导致，修改配置文件仍然采用LwIP系统自带的内存池分配方式，运行一段时间后，任务仍然“阻塞”。

**怀疑4：**由于LwIP的各个功能的超时是由移植函数sys\_arch\_mbox\_fetch()驱动，根据文档对该函数进行进一步检查和修改完善，运行一段时间后，任务仍然“阻塞”。

**分析原因：**以上所有的步骤结果都是一样：任务阻塞在lwip\_netconn\_do\_connect()函数中以下语句：

UNLOCK\_TCPIP\_CORE();

sys\_arch\_sem\_wait(LWIP\_API\_MSG\_SEM(msg), 0);

LOCK\_TCPIP\_CORE();

而该信号量的释放是在慢定时器tcp\_slowtmr(void)函数中，由定时器驱动超时、正常等情况下的信号量post，经过写入log发现慢定时器tcp\_slowtmr(void)停止运行，而LwIP的系统定时器tcp\_tmr(void)也是停止（没有）运行，经过分析源程序，发现代码如下调用：

tcp\_slowtmr()----->tcp\_tmr(void)----->tcpip\_tcp\_timer()----->tcp\_timer\_needed----->被如下宏定义调用：

#define TCP\_REG(pcbs, npcb) \

do { \

(npcb)->next = \*pcbs; \

\*(pcbs) = (npcb); \

tcp\_timer\_needed(); \

} while (0)

而TCP\_REG被TCP状态机处理函数tcp\_process(struct tcp\_pcb \*pcb)调用，也就是说TCP的定时器并不是每时每刻运行，而是在LwIP系统状态机切换时，根据实际需要进行注册定时器链表，以时间驱动向前执行，比如超时判断、窗口因子调整等等。

在tcp\_connect(struct tcp\_pcb \*pcb, const ip\_addr\_t \*ipaddr, u16\_t port,tcp\_connected\_fn connected)函数中：

有一个定时器链表注册TCP\_REG\_ACTIVE(pcb)语句，在建立连接过程中，启动定时器。---如果因为状态转换而触发某个条件导致这个语句如果没有执行，那么就会导致定时器不执行，也就是信号量不能释放post，就会出现lwip\_netconn\_do\_connect()函数阻塞在信号量的情况。

而定时器链表是如何驱动的呢？这是由LwIP内核主任务进程tcpip\_thread()中执行，通过函数tcpip\_timeouts\_mbox\_fetch(sys\_mbox\_t \*mbox, void \*\*msg)中的移植函数sys\_arch\_mbox\_fetch(mbox, msg, sleeptime)驱动执行，因此该函数移植的质量决定着系统核心主进程是否正常运行。

按照说明对sys\_arch\_mbox\_fetch(mbox, msg, sleeptime)移植函数进行进一步检查修改，但是测试结果表明，该函数不存在问题，但是仍然会出现任务阻塞的情况，因此可以初步得出结论如下：

由于在系统状态机转换过程中，偶然触发的条件导致定时器链表没有执行或者注册失败，导致不能进行时间超时判断，也就是在内核动作不能得到进一步的执行，最终会出现如下现象：

1. 、任务会阻塞在信号量上，因为没有注册定时器链表而不能判断超时而post信号量。
2. 、LwIP在长时间运行过程中，可能在状态切换过程中，没有注册定时器链表/注册定时器链表失败，超时机制不能正常运行，从而导致某些分配的内存不能回收，出现内存泄漏。

最终结论：LwIP内核存在的BUG。

解决方案：等待下一个版本，期待进行修复。(可是先回溯到2.0.3)