|  |  |
| --- | --- |
| Bug ID（缺陷标识符） | HBASE-3449 |
| Bug Report Page（缺陷报告页面） | https://issues.apache.org/jira/browse/HBASE-3449 |
| Bug Type（缺陷类别） | 本地通信死锁 |
| Bug Description（缺陷描述） | 在HBASE-3449中，如图1所示，节点master使用消息队列msgQueue保存所有META服务器发送过来的消息包括shutdown消息，且允许至多2个线程同时处理这些消息。假设存在META服务器server1、server2和server3。其中server1托管表META1和ROOT，当它关闭时会向master发送shutdown消息，master调用消息处理线程thread1处理该消息，thread1的处理方式是先将ROOT重新分配到server2，再将META1重新分配到server3，thread1此时正等待server2向master返回成功接管ROOT的消息。服务器server2在ROOT还没有成功分配到自己时就关闭，当它关闭时向master发送shutdown消息，master启动消息处理线程thread2处理该消息，thread2的处理方式是将ROOT重新分配到server3，并开始等待server3向master返回成功接管ROOT的消息。服务器server3在ROOT还没有成功分配到自己时就关闭，当它关闭时向master发送shutdown消息，但master至多只允许2个线程同时处理消息，故它无法调用线程处理该消息。此时，ROOT表仍然没有被成功分配到特定的服务器。这样在master中，thread1和thread2都在等待关于ROOT表的重分配结果的消息，而这个消息只有当thread1或thread2能够成功处理server3发送过来的shutdown消息后才能发送。这就造成位于同一进程内的两个线程thread1和thread2在执行任务时等待thread1或thread2执行完毕其他任务后才能发出的消息，从而构成本地通信死锁。 |
| Bug Diagram（缺陷示意图） | 图1 HBASE-3449本地通信死锁发生过程示意图  Fig.1 A diagram to show how the local communication deadlock HBASE-3449 happens |