和金轩/谢文/赖亿商量， dbproxy 线程分发结构方案实施

前端的处理方式：
基本一致：主要思路都是一个 con 由一个线程处理其状态机中所有的状态。改进点就是：主线程向 worker thread 发送的过程中，
是轮训发送，还是说惊群让线程去抢的问题。考虑到任务分配的均匀性，可能会选择惊群，但是这个均匀只能是登陆 client 的均衡。感觉哈。。。

后端连接池的处理方式：
1. 连接池采用不注册 libevent 的机制，对 2 种异常做如下处理
a. 连接池长时间释放：每个连接加入最近执行时间，另外一个线程去轮询看是否时间太长了，太长了就释放
b. mysql server 端主动关闭连接：由前端来负责处理这种情况，前端取得连接先判断是否连接可用，如果不可用，再取一个连接，如果再不可用就触发重新建立连接。
如果想做的更好点，前端发生这种情况通知一个线程去扫描一下连接池，剔除这种连接不可用的情况

备注：
为啥不用连接池采用不注册 libevent 的机制?
会有 libevent 的 1 个事件注册到 1 个线程上(在回调函数中用了 mutex)，另一个线程来删除这个事件时会死锁的情况，因为我删除时需要 mutex 来保护。

==========
发件人: Hou JinXuan(运维中心)
发送时间: 2013 年 7 月 16 日 9:55
收件人: Lai Yi(运维中心); Xie Wen(运维中心); Fan Zhen(运维中心)
抄送: Li YongZhi(运维中心)
主题: 答复: dbproxy 线程分发结构改造会议纪要

大家好：

方案选择：
前端的处理方式：
基本一致：主要思路都是一个 con 由一个线程处理其状态机中所有的状态。改进点就是：主线程向 worker thread 发送的过程中，是轮训发送，还是说惊群让线程去抢的问题。考虑到任务分配的均匀性，可能会选择惊群，但是这个均匀只能是登陆 client 的均衡。感觉哈。。。

后端连接池的处理方式：
1. 连接池除了按照用户划分，还需要按照线程划分。
缺点是：连接复用的粒度变得很小了。做个自适应的连接池对于线程的分配策略还是比较麻烦的；此外还需要对连接池的结构作调整。
优点就是：线程池不会再出现死锁的问题了。

2. 提前 5-10s 使的连接池中的空闲时间过长的连接不可用。方法为：记录每个连接的空闲开始时间在超时前 5-10s 内，连接都认为是不可用的。不可用连接可以等待超时去关闭，当让也可以重新启动一个线程去轮训关闭。个人倾向于超时关闭，简单。
但是这两种方式都有一个问题：即不能很好地处理 mysql server 端主动关闭连接的情况，在这种情况下超时事件处理线程接收到的是 EV_READ 事件其到来的时间是随机的随时的，是通过提前 5-10s 无法避免的。
3. 线程池中共享的变量是 entry, 现在产生 core dump 的原因是 entry 变量被释放但是事件处理线程不知道，而我们没有找到 kpi 来获知一个内存被是否被释放而后又被使用。因而就想着会引入垃圾队列，可以保存内存已经被释放的现场也可以避免内存重新分配导致的内存非法访问。因而我们就会在连接使用、连接超时释放及后端关闭导致连接释放的时候，将 entry 放入一个垃圾队列。同时记录放入垃圾队列的时间，放入垃圾队列一段时间后 5-10s 后才释放 entry 内存。
优点：基本避免了原来的因为连接池公用引起的 core dump 的原因。
缺点：与前面的方法比代码相对多些，再有内存使用较多因为没有随时释放。还有就是何时那个时间（5-10s）不好选择。但是个人认为稍微延后加锁的机制就能避免 core dump.
我和谢文比较倾向于方案 3.

谢谢
侯金轩

==========
发件人: Lai Yi(运维中心)
发送时间: 2013 年 7 月 14 日 21:30
收件人: Lai Yi(运维中心); Xie Wen(运维中心); Hou JinXuan(运维中心); Fan Zhen(运维中心)
抄送: Li YongZhi(运维中心)
主题: 答复: dbproxy 线程分发结构改造会议纪要

我能想到的一些细节，大家参考如下
1. 连接池中连接释放，我觉得对连接记录一个时间，然后线程定期扫描检查起个连接池连接是否需要释放就行了，没必要使用 libevent 的 timeout 触发。如果用 libevent 的 timeout 好像还是很麻烦，即使提前设置 disable。
2. 360dbproxy 的建立连接时是轮询指定线程，我觉得如果想优化的话还是广播一下，还是按 dbproxy 思路让线程去抢的方法更加合理，我们可以讨论一下（当然建立连接后 network_mysqld_con 和一个线程绑定）

发件人: Lai Yi(运维中心)
发送时间: 2013 年 7 月 14 日 12:30
收件人: Lai Yi(运维中心); Xie Wen(运维中心); Hou JinXuan(运维中心); Fan Zhen(运维中心)
抄送: Li YongZhi(运维中心)
主题: 答复: dbproxy 线程分发结构改造会议纪要

==========

发件人: Lai Yi(运维中心)
发送时间: 2013 年 7 月 14 日 12:12
收件人: Xie Wen(运维中心); Hou JinXuan(运维中心); Fan Zhen(运维中心)
抄送: Li YongZhi(运维中心)
主题: dbproxy 线程分发结构改造会议纪要

dbproxy 线程分发结构改造会议纪要
参与人：谢文，金轩，赖亿
开会时间 ： 2013，7.14 （周日） 10：30—12：00
主题：线程分发结构改造会议纪要

1.为啥需要修改
a. 这周在做 dbproxy 的打压测试是，周二发生 coredump,线程死锁，
修正了部分代码配合升级 libevent 库（可惜升级的 libevevent 只有 alpah 版本），线程死锁解决。
周五又发生线程自己锁自己的，事件重复添加的问题，还没有解决

问题根源是因为 network_mysqld_con（跟踪用户连接状态机核心结构） 在状态机变化时需要被多个线程共享，
加上现在稳定版本 libevent 功能有缺陷导致。我们现在被迫升级下一个 alpha 版本 libevent 库。

b.另外一个原因是，以后 dbproxy 功能越来越复杂，如果现在都很郁闷，以后 dbproxy 的扩展性很成问题。

2.如何修改和代价
仿照现在开源 360 的 dbproxy 的做法，建立连接时， network_mysqld_con 和一个线程绑定，以后在状态机变化时就在这个线程中处理，
这样就不需要加锁了。

代价：a. 初步预计 2 周时间修正代码，确定时间需要下周二设计讨论完给出。
b. 建立连接后，这个连接就和线程绑定了，负载均衡的粒度变差。
不像以前 （每个连接+连接中状态机）和线程绑定。但是我们认为可以接受

3.实施计划：
1.谢文为主看 360dbproxy 写出设计文档, 金轩同时写一个设计文档， 周二一起讨论 pk。
2. 讨论完了一起，谢文开始写代码