复用解决方案

彭程 1352905

Q:用户登录后始终在线，考虑低宽带／不稳定网络

——长连接心跳机制

——消息不遗漏

——消息不重复

——消息压缩

… …

Solution:

当用户登录某个系统后，如果始终保持在线状态，而其网络又处于一种不稳定的状态，这时我们需要采取一定的措施来实时确认网络的连接状态，同时可以将需要传送的消息以更小的代价（消息压缩等）进行传递，查阅过大量的相关资料后，我整理得到以下可供参考解决方案：

1.应用心跳机制（heartbeat mechanism）

心跳机制是定时发送一个自定义的结构体，让对方知道自己还活着（在线），以此来确保连接的有效性的机制。



(图片来自百度百科)

具体说来，网络中的接受和发送数据都是实用操作系统中的SOCKET实现。如果用于标示客户端请求和服务器请求的套接字断开，那么发送数据和接受数据的时候就会存在问题。通过心跳机制，我们可以确保对方“在线”。以确保连接的有效性。

我们通过心跳包来作为传递信息的“介质”。所谓的心跳包就是客户端定时发送简单的信息给服务器端告诉它我还在而已。 服务端收到后回复一个固定信息如果服务端几分钟内没有收到客户端信息则视客户端断开。比如有些通信软件长时间不使用，要想知道它的状态是在线还是离线就需要心跳包，定时发包收包。发包方：可以是客户也可以是服务端，看哪边实现方便合理。一般是客户端。服务器也可以定时轮询发心跳下去。心跳包之所以叫心跳包是因为：它像心跳一样每隔固定时间发一次，以此来告诉服务器，这个客户端还活着。事实上这是为了保持长连接，至于这个包的内容，是没有什么特别规定的，不过一般都是很小的包，或者只包含包头的一个空包，服务器如果在一定时间内收不到客户端发送过来的反馈包，那么就可以认定掉线了，在我们的具体环境中，用户的登录就是无效的了，此时需要将用户的登录注销。

这里主要提出两种解决方案：

1.1通过两结点甚至多结点集群心跳机制来预防单点失效问题

这点主要借鉴 参考4 ，通过利用高可用集群系统中冗余的可脱离的部件，在系统出现单点故障（SP0F-Single Point Of Failure）的情况下降级运行，从而提高系统的可用性。

我们以两结点集群的心跳机制为例，说明其工作过程：

1. 主状态启动进程后，会创建一个时钟。主状态进程根据时钟信号，周期性地把本地节点状态信息打成数据包，写入FIFO.
2. 控制进程从FIFO中读出数据包，同时写入主状态管道和通讯管道。通讯管道用于心跳传输介质写进程和控制进程之间的通讯，每种心跳传输介质分别有一个通讯管道。
3. 心跳传输介质写进程从通讯管道中独处数据包，通过心跳传输介质发送给另一个节点。
4. 心跳传输介质读进程从传输介质读取另一个节点发送来的数据包，写入主状态管道。
5. 主状态从主状态管道获得两个节点的状态信息，利用时间记录判定节点是否实效。

多节点的集群心跳机制于此类似，通过分析，个人认为虽然这是这是基于Linux的高可用心跳机制，但和我们的登录环境一样，都依靠网络来传输心跳信息，是一套可以复用的方案，可以用来防止网络传输可能产生的单点失效问题，同时可以根据相关的时间值判断节点状态变化。

1.2通过改进心跳包机制来提升系统可靠性和可用性

这点主要借鉴 参考5 ，这种方案从另一个维度——对心跳包机制的改进来提高系统可用性。

具体说来，就是在客户端与服务器设计不同的心跳机制，通过在服务器与客户端之间传递心跳包数据，判断服务器与客户端之间的连接状况，并在网络拥塞时通过连续请求来实现重新连通，从而保证系统在网络拥塞时能够有效连通，在网络断开时提示用户。

个人认为这也是一套可以复用的系统，通过心跳包来检测网络异常中断，异常中断主要有两种情况：网络连接的无力设备出现问题，从而导致连接中断；网络堵塞导致数据无法正常发送，从而服务器或客户端认为对方长期没有响应，判断为连接中断。具体设计可以参考改模型来设计专门针对客户登录的网络心跳包。

2.消息压缩

在低宽带或不稳定网络的情况下，很可能没有机会进行大量的数据的传送，这是消息的压缩就是一个不得不考虑的问题，这也是我分析的第二个层面。

对于任何形式的通信来说，只有当信息的发送方和接受方都能够理解编码机制的时候

压缩数据通信才能够工作。例如，只有当接受方知道这篇文章需要用汉语字符解释的

时候这篇文章才有意义。同样，只有当接受方知道编码方法的时候他才能够理解压缩

数据。拥有双方都理解的编码机制，才能在此上进行非破坏性消息压缩甚至破坏性的消息压缩。通过查阅资料，参考以下两种方案：

2.1 route压缩

这一点主要借鉴 参考6 ，在实际编程中，为了减少数据传输带宽的消耗，提高传输效

率，pomelo提供了对消息的压缩，包括基于字典的对route的压和基于protobuf的

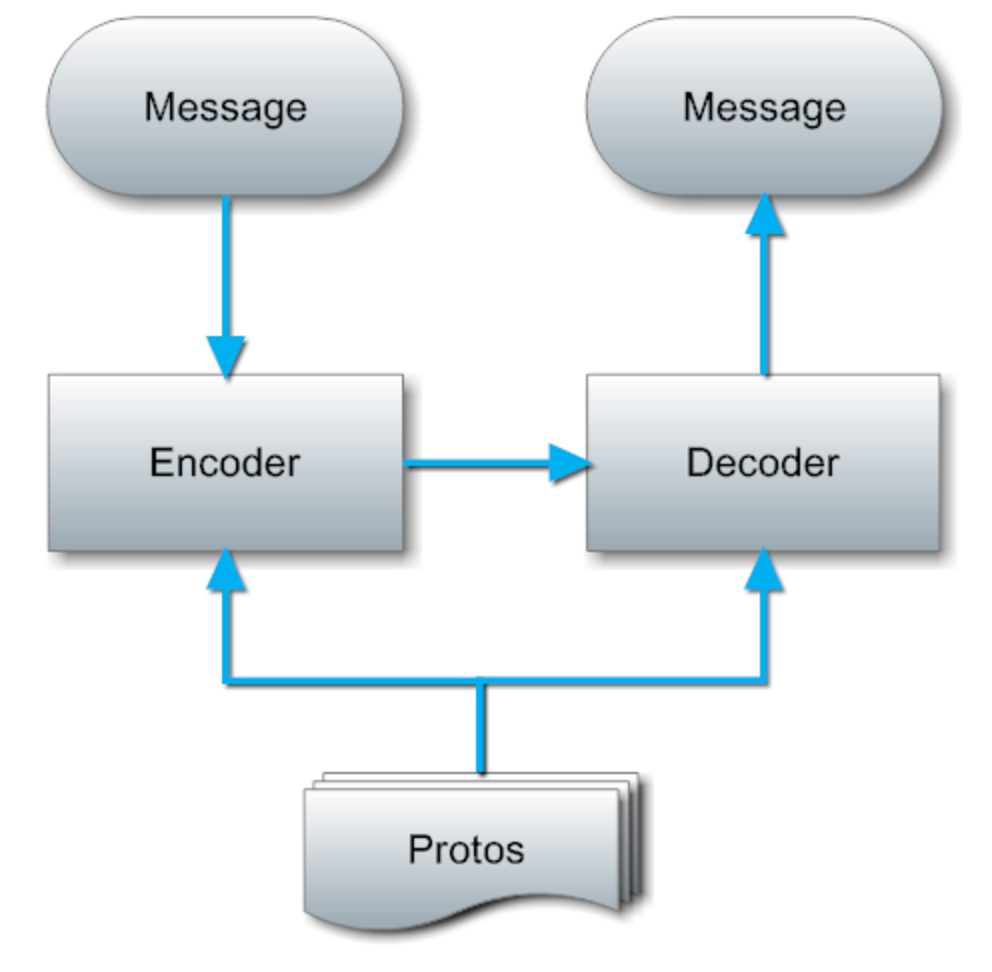
对具体传输数据的压缩。

个人认为这完全是一套可以复用的数据压缩机制，通过对数据的压缩提高了带宽的有

效数据利用率，使得在一些带宽以及流量敏感的环境中，pomelo能够更好地工作。

只要protobuf编／解码器设计得当，可以比较好的复用于极端网络情况,其复用模型如

下。



（图片来自GitHub）

2.2对HTTP节点和SOAP节点使用压缩

这一点主要借鉴 参考7 ，可以在发送和接收消息时，将 HTTP 和 SOAP 节点配置为

使用 HTTP 压缩和解压缩。而这些节点可配置为压缩请求消息、接受和解压响应，

以及解压输入消息。

这里从另一个维度（协议节点）来考虑数据压缩问题，每一个具体压缩节点都可以参

考具体的数据模型，个人认为这也是一种非常好的研究方向。

以上便是个人总结过后，基于软件复用的角度对该问题的看法。

参考：

1. <https://en.wikipedia.org/wiki/Heartbeat_(computing)>（wiki heartbeat）
2. <http://baike.baidu.com/subview/4372209/4372209.htm>（百度百科 心跳机制）
3. <http://www.vldb.org/conf/2005/papers/p1079-johnson.pdf>（A Heartbeat Mechanism and its application in Gigascope）
4. <http://kreader.cnki.net/Kreader/CatalogViewPage.aspx?dbCode=CJFQ&filename=JSGG200401020&tablename=CJFD2004&compose=&first=1&uid=WEEvREcwSlJHSldTTGJhYlNPS3ZvdjRqZnFDbTJEOTJ5djR5Z2k1U0tYYTdGVHovb0YybzdLRGJMREVLTEtCVkhiZz0=$9A4hF_YAuvQ5obgVAqNKPCYcEjKensW4IQMovwHtwkF4VYPoHbKxJw>!!

（Linux高可用集群心跳机制研究）

1. <http://kreader.cnki.net/Kreader/CatalogViewPage.aspx?dbCode=CJFQ&filename=JSJY200802029&tablename=CJFD2008&compose=&first=1&uid=WEEvREcwSlJHSldTTGJhYkdRbHpDOTdvdFR2THR1Y0NwSnYrdDY3dDNvYVZ5RUU2cTRscDRzMzNmV1p1ZWJyS2s4VT0=$9A4hF_YAuvQ5obgVAqNKPCYcEjKensW4IQMovwHtwkF4VYPoHbKxJw>!!

（基于改进心跳包机制的整流远程监控系统）

1. <https://github.com/NetEase/pomelo/wiki/%E6%B6%88%E6%81%AF%E5%8E%8B%E7%BC%A9>（GitHub 消息压缩）
2. <http://www.ibm.com/support/knowledgecenter/SSMKHH_9.0.0/com.ibm.etools.mft.doc/bc19010_.htm?lang=zh>（IBM压缩算法）
3. <https://zh.wikipedia.org/zh/%E6%95%B0%E6%8D%AE%E5%8E%8B%E7%BC%A9>（wiki 数据压缩）