在给cowboy做压力测试时遇到一些问题，总结列出。

1、修改ulimit ulimit -SHn 1000000（linux文件句柄数socket也是文件句柄），如果不修改客户端connect socket并发数量受限。

2、服务器客户端出现过多time\_wait，导致端口无法重用（端口是有限的只有65535个）， 服务器在处理客户端请求的时候，如果你的程序设计为服务器主动关闭，那么你才有可能需要关注这个TIMEWAIT状态过多的问题。如果你的服务器设计为被动关闭，那么你首先要关注的是CLOSE\_WAIT**。**解决方法：

发现系统存在大量TIME\_WAIT状态的连接，通过调整内核参数解决，

 vi /etc/sysctl.conf

 编辑文件，加入以下内容：

     net.ipv4.tcp\_syncookies = 1       表示开启SYN Cookies。当出现SYN等待队列溢出时，启用cookies来处理，可防范少量SYN攻击，默认为0，表示关闭；

net.ipv4.tcp\_tw\_reuse = 1           表示开启重用。允许将TIME-WAIT sockets重新用于新的TCP连接，默认为0，表示关闭；

net.ipv4.tcp\_timestamps = 1

     net.ipv4.tcp\_tw\_recycle = 1          表示开启TCP连接中TIME-WAIT sockets的快速回收，默认为0，表示关闭。

net.ipv4.tcp\_fin\_timeout = 30  对于本端断开的socket连接，TCP保持在FIN\_WAIT\_2状态的时间。对方可能会断开连接或一直不结束连接或不可预料的进程死亡。默认值为 60 秒。

然后执行/sbin/sysctl -p 让参数生效。

3、客户端产生etimedout报错。

Linux内核协议栈为一个tcp连接管理使用两个队列:一个是半链接队列（half open(syn queue) 用来保存处于SYN\_SENT和SYN\_RECV状态的请求），一个是全连接队列（accpetd队列）（用来保存处于established状态，但是应用层没有调用accept取走的请求）， 服务器的连接队列满掉后，服务器不会对再对建立新连接的syn进行应答，所以客户端的 connect 就会返回 ETIMEDOUT（这里无法解释太清楚，具体需要看tcp内核处理逻辑）。

半链接队列 是由内核参数tcp\_max\_syn\_backlog 控制，可以通过 ehco Value > /proc/sys/net/ipv4/tcp\_max\_syn\_backlog 更新，或者 在/etc/sysctl.conf中添加net.ipv4.tcp\_max\_syn\_backlog=Value，然后执行sysctl -p使其生效。

全连接队列 大小为 min(somaxconn, backlog)，backlog参数为listen自带参数，somaxconn是内核参数，用于限制backlog，当backlog大于somaxconn时则取somaxconn的值。somaxconn可以通过echo **Value** >   /proc/sys/net/core/somaxconn 更新或者在/etc/sysctl.conf中添加net.core.somaxconn = Value，然后执行sysctl -p使其生效。

4、当把**tcp\_max\_syn\_backlog，**backlog，somaxconn参数数值调整较大后，加大客户端并发量，依然会出现etimedout报错，说明accept处理比较慢，可以通过加大accept进程数量和加快连接处理速度提高效率。

5、cowboy里accept处理一个连接时，会调用ranch\_conns\_sup:start\_protocol，这里是同步处理，会发消息给ranch\_conns\_sup进程，并等待返回，ranch\_conns\_sup进程收到消息，创建业务处理进程（http处理模块），绑定socket到业务处理进程，返回消息给accept进程，处理下一条消息。当ranch\_conns\_sup 处理连接比较慢时，会导致消息积压，导致accept阻塞。可以考虑将ranch\_conns\_sup:start\_protocol做成异步，不等结果返回，或者多起几个ranch\_conns\_sup,从而提高处理速度。

6、优化业务处理模块（http处理模块）。

tcp总结：

1、服务器影响：服务器硬件配置和内核参数、tcp参数设置。内核参数和tcp参数主要涉及连接队列的处理。服务器调用listen后系统内核开始接受连接，listen有两个队列，半连接队列（未完成3次握手，处于SYN\_SENT和SYN\_RECV状态 的请求）和全连接队列(完成了3次握手，处于established状态 的请求），这两个队列的长度是受内核参数和tcp参数控制的（tcp\_max\_syn\_backlog ,backlog,somaxconn ）。调用accept会从全连接队列取socket连接。当连接队列配置的足够大时候就要看accept处理连接队列的效率。

2、accept处理连接队列的效率 。accept处理效率有两个因素决定：accept进程数量，accept处理socket连接效率。关于accept进程数量可以通过创建更多accept进程提高。accept进程处理效率看业务处理能力。当配置好足够的accept进程数量后，系统瓶颈就在socket连接处理效率上了。

3、socket连接处理效率提升。这里就主要看也业务处理效率了。当accept取到一个连接后应尽快将连接绑定到新的进程上（这里会阻塞 ），继续进行下一个连接。

4、将连接绑定到新进程上就是业务处理模块了，到此整个tcp处理完成。这里会成为新的瓶颈。

个人认为把tcp效率调到一定高度，最终瓶颈还是在业务处理速度上。当tcp连接很高时，内存，cpu,io都会出现瓶颈，这时候主要优化就在这些上面了。