# 异数OS TCP协议栈测试（三）--长连接篇

## 本文来自异数OS社区

github： <https://github.com/yds086/HereticOS>

异数OS社区QQ群： 652455784

## 异数OS TCP长连接技术简介

说起长连接，则首先要谈对C10K的理解与认识，异数OS认为传统系统中C10K问题主要如下：

1. **传统OS 线程切换代价很大**，这使得多线程阻塞IO性能达到1W时，线程切换将占满系统负载，而异数OS线程切换能力为80M,所以可以使用这种方案。
2. 为了避免线程切换带来的问题，主流操作系统使用队列技术(iocp epoll)来批量成倍提升IO性能，每次线程切换完成10个甚至100个IO操作，但这类技术反而造成了更严重的C10K问题，使用这一技术，每个链接必须要提供中间传输缓存，链接多了后，**队列会很长，延迟会很高，加重了系统负担**，由于这类技术都是非阻塞异步IO,因此无法做可靠的QOS控制，雪崩时问题更加严重。
3. TCP协议栈没有为海量链接做优化，其常见的流控拥塞算法都只针对链接自身的IO性能，并不考虑上层应用实际需求，以及其他链接的IO需求，在这种情况下**TCP检查网络拥塞的算法将没有任何意义**，实际测试发现如果每链接IO不做控制，1000链接都很难上去，正确的做法是将拥塞控制算法提高到系统层以及应用层由系统QOS任务调度以及应用需求共同来实现。
4. **协议栈中间缓存太多，拷贝动作较多,**传统OS每链接协议栈需要4K以上内存，实做一个1000W链接的系统一般要60G内存起步，这直接增加了系统负载，降低了系统可用性。
5. **协议栈IO性能不足，**由于传统操作系统IO性能约束，实做的协议栈只能提供10-40W的IO性能，并且不能多核扩充，在这样的情况下，做1000W链接的应用可用性会受到极大的限制，比如微信使用这一类技术就只能用于心跳链接活跃检查。

一些号称实做了C10M的技术方案从本质上没有解决上述问题，仅仅靠硬件技术堆硬件配置来表面实现，但实际上却没有任何可用性。

## 异数OS TCP长连接技术演变

2015年，异数OS考虑开始使用同步阻塞IO的方案来**解决iocp不能做QOS,队列过长的问题**，异数OS使用任务调度来控制每链接的IO提交性能，从而减少了中间缓存开销，降低了海量并发下iocp队列深度，降低对宿主操作系统的负载压力，从而实现了1000W链接12W的消息推送性能。

2017年，由于发现宿主操作系统协议栈的能力约束，异数OS决定抛弃宿主操作系统协议栈，以及IO技术，开始**自主研发TCP协议栈**，实做0中间缓存，1次中间拷贝的技术，从而达到每链接300字节占用，并发容量相对异数OS 2015,提高15倍，IO性能提升100倍，且可以多核扩充，大大提高了海量并发环境的系统可用性。

## 测试目标

TCP 长链接IO性能测试，Client Server都采用单线程半双工模式，创建600W客户端（本机测试相当于1200W链接），链接服务端后做循环ECHO,测试ECHO IO性能，IO延迟，每链接性能稳定性。

## 基本测试环境

VMware 12

异数OS宿主操作系统 debian 8 64位

CPU : NUC i3 2.6G 双核

内存:5GB

相关参数

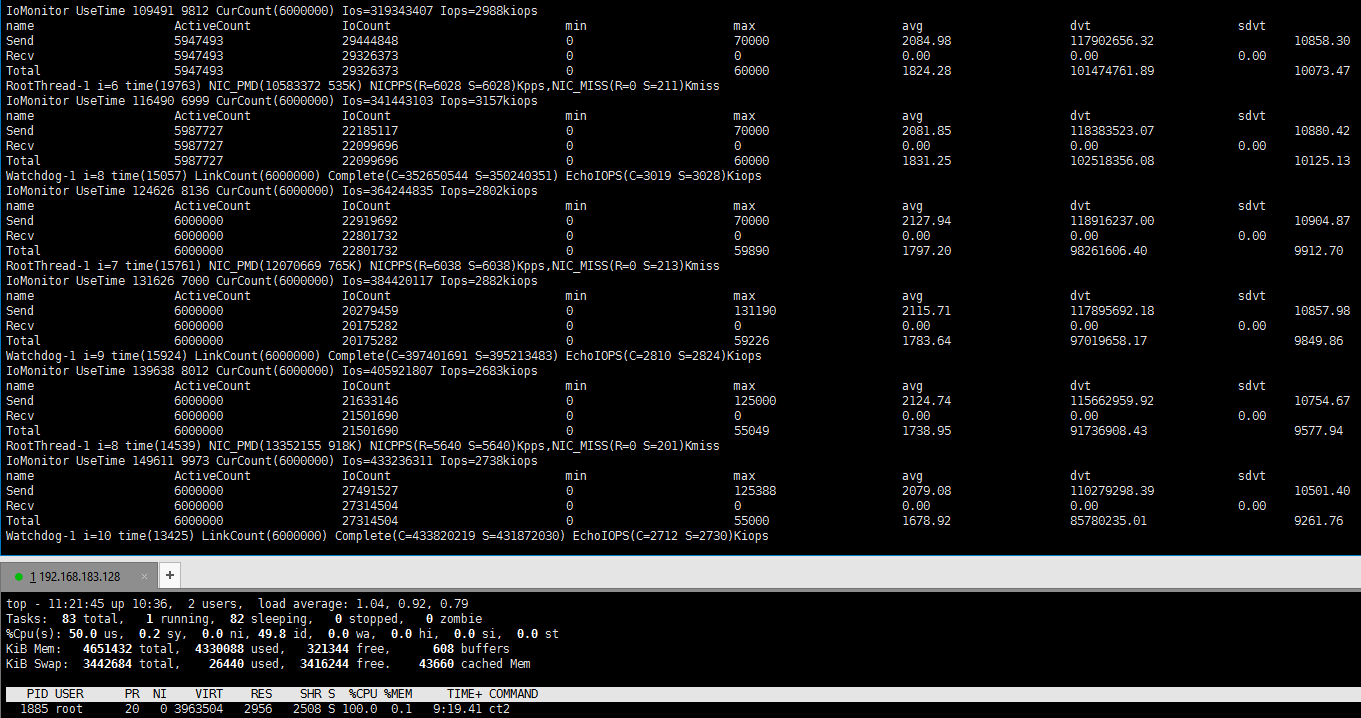
1. 带包头200字节负载，不带crc checksum, 无丢包,无硬件延迟情况。

2. TCP协议栈使用均衡IO调度策略。

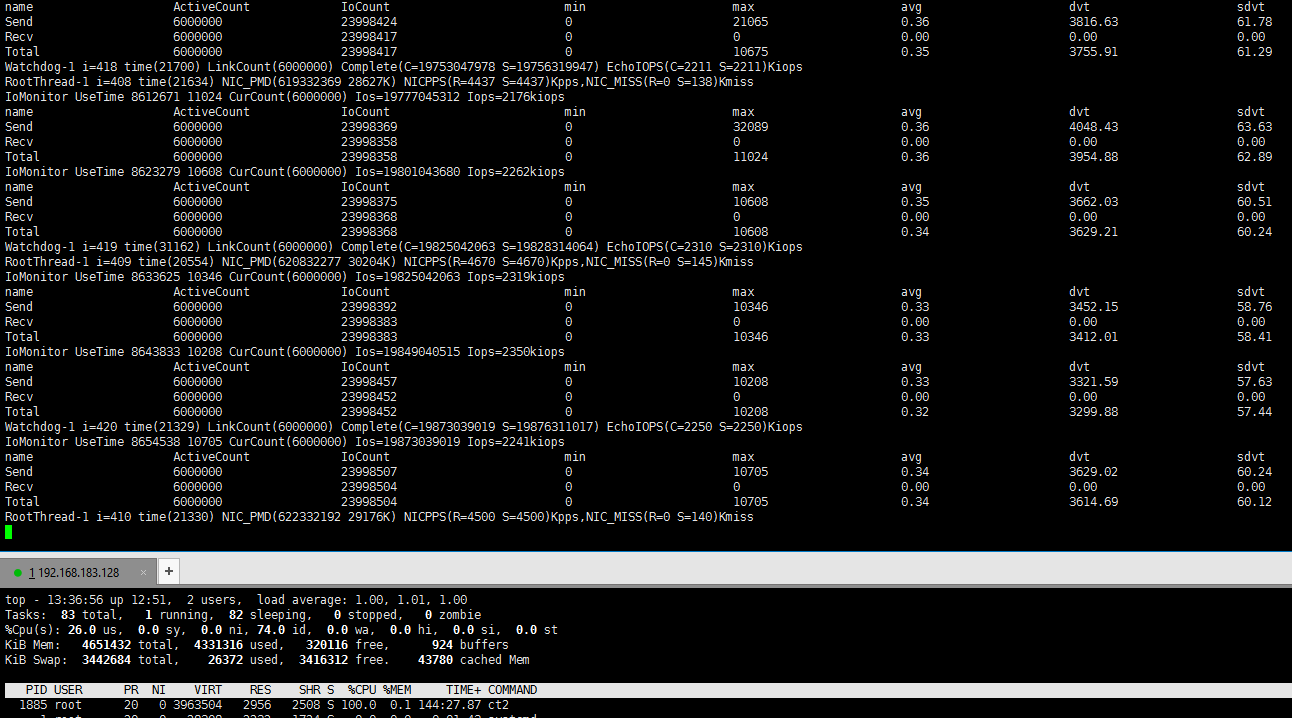
## 测试方案一 (单核)

在同一个CPU核上创建Server,600W个Client, 以太层使用异数OS软件交换机本地核定向转发。

启动前期：



启动后期（2个多小时后）：



总计ECHO IOPS 为2.3M ,软件交换机包交换能力9Mpps，由于Client占用60%的负载，软件交换机占用20%负载，所以预计真实环境中最大可达到6.0M左右的ECHO能力，IO延迟方面，在系统启动阶段由于大量链接的建立，每链接IO需求不稳定，系统QOS IO均衡并没有显著的发生作用，平均延迟达到了2秒，一些链接任然有饥饿现象，出现上百秒才响应的情况，在稳定链接后期,每链接IO需求稳定后，IO延迟下降，饥饿现象得到缓解，但还是有10s延迟的链接出现。

## 总结

由于只使用了TCP协议栈的任务均衡调度控制方案，因此每链接IO质量在应用负载不均的情况下还是不能得到及早的控制，这个问题后面会由应用系统的QOS来解决，比如mqtt等专业的APP QOS控制,下面是几种主流系统的海量链接平台能力性能对比，数据来自官网以及第三方测试，可比性可能不高，但也可做参考估算，读者如有其他海量并发的技术测试也欢迎提供

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 对比项目 | 异数OS 2018 | 异数OS 2015+Win7 | 异数OS 2015+Win10 | 360push | Whatsapp |
| CPU占用数量 | 1 | 16 | 16 | 24 | 12 |
| 内存占用 | 4G | 64G | 64G | 256G | 128G |
| 实现的链接数 | 1200W | 1000W | 300W | 100W | 300W |
| 使用的技术平台 | 寄宿Linux下+异数自主协议栈 | 寄宿win下+iocp | 寄宿win下+iocp | Go+epoll | Erlang+epoll |
| 测试内容 | ECHO 推拉 | 仅推 | 仅推 | 仅推 | 仅推 |
| 推送性能 | 4.5M | 12W | 40W | 2W | 12W |
| 折算的IO性能 | 9M | 12W | 40W | 2W | 12W |
| 已知问题 | 缺乏经费支持 | 缺乏经费支持 | 缺乏经费支持 | 容易宕机，要反复重启 | 特制应用平台，不同用。 |