吞吐量（TPS）：系统单位时间内处理的请求数及任务数

系统延迟/响应时间：系统在处理一个请求或者一个任务间的延迟

【服务器的性能】

服务器主要硬件：CPU，内存，硬盘，网卡（主要约束项任何一个达到最大值则服务器性能就会受到影响）

CPU：CPU系统使用率，CPU用户使用率

CPU系统使用率？

CPU用户使用率？

内存：进程占用总内存，分析内存趋势图（内存回收机制是否正常，是否存在内存泄露的情况）

磁盘：磁盘I/O操作时间

网卡：网卡读写包量，网卡读写速度

【工具使用】

参数设置

起始人数：

增量：

阶段时间：

最大人数：

超时时间：

说到底就测试场景设计

目的：定位应用的性能瓶颈，并提出适当的优化方案

压力场景设计的方式：

常见的场景类型：

1.单场景

2.复合场景

性能测试过程：

需求调研，环境准备，脚本开发，数据预埋，场景设计，场景执行，应用监控分析，瓶颈定位，瓶颈修复，回归测试，结果对比与整理

测试模型：延时策略，运行时长，加压策略，并发用户数量，执行时长，终止方式，资源监控策略

测试模型和测试指标是进行场景设计的前提和基础。

被测系统不同，可能测试指标也有差异。

测试指标：

web类应用：在线用户数，最大并发用户数，最优并发用户数，平均响应时间，目标TPS

接口调用类：平均响应时间，目标TPS

复合场景压力测试怎么设计呢？

一般情况下，依据业务流程。如：登陆-加入购物车-下单-付款

测试模型如何确定？

未上线的系统：

已上线的系统：依据实际业务来确定数据比例，在此基础上成倍增加用户请求数

压测对象选取：可选取占比较大的服务进行压测

测试指标

最大并发用户

最优并发用户

梯度加压

并发用户数：

目标TPS\*交易平均响应时间=并发用户量？

最主要的仍是在测试报告的分析

加载策略：

同时加载，指定时间间隔，梯度加压

运行时间：

基准测试场景：10M

负载测试场景：10-30M

混合测试场景：10-30M

稳定性场景：8-24h

扩展性场景:10-30M

Linux CPU :

perf工具2019/4/14 16:57

perf record -e cpu-clock -g -p 2388

perf report -i perf.data

valgrind --tool=memcheck --leak-check=full./test