

# Linux性能分析工具二

丁静 dingjingdjdj@gmail.com

# USE-性能分析方法

- \* 使用率

在一定时间间隔内，忙于处理请求的线程平均数目

- \* 饱和度

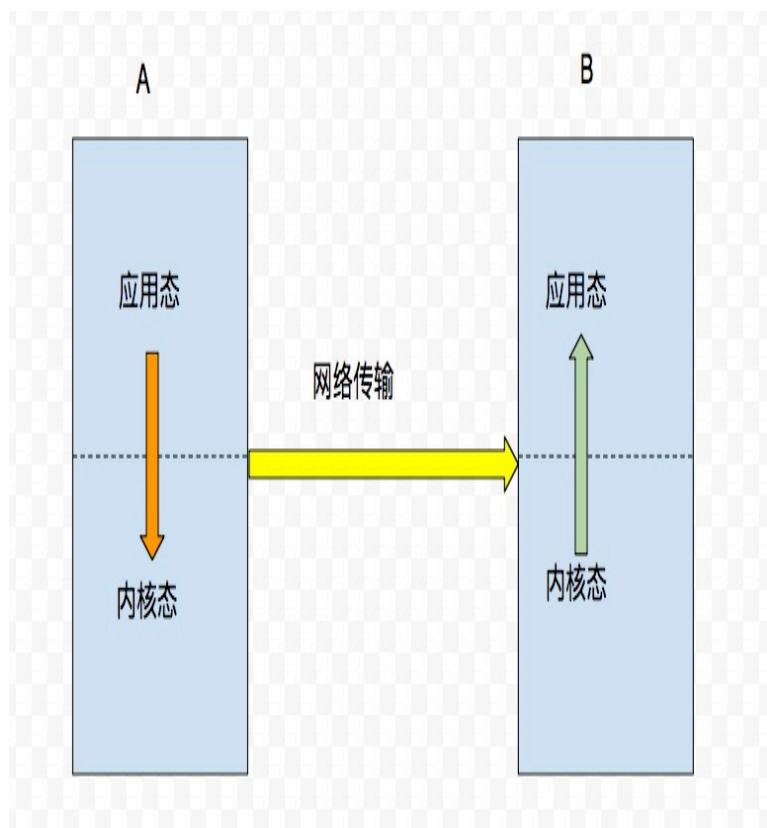
在一定时间间隔内，请求队列的平均长度

- \* 错误

处于某种原因，请求拒绝或者失败

`dmesg | tail`

# 确定性能路径



tcpdump wireshark 网络分析

包时间点，应用层日志

丢包分析，网络错误

tcp window 滑动窗口

nagle 算法， delay ack

# CPU优化准则

- \* 减少不必要的工作，减少CPU使用
- \* 使用缓存，空间换时间，减少计算逻辑，LRU算法
- \* CPU密集型的考虑简单有效的算法，结合CPU特性，分支预测，局部性等
- \* 优化执行最频繁的代码路径
- \* 减少复杂数据结构的使用，频繁使用的情况下，结合业务场景，创造新轮子

# CPU优化步骤

- \* 使用perf生成火焰图，分析热点路径
- \* 确定热点路径耗时情况，耗时波动，执行次数
- \* 耗时长的堆栈路径，输入参数
- \* 系统调用分析

# CPU调度分析

- \* `sudo perf sched record -a sleep 2`
- \* `sudo perf sched latency -s avg`
- \* `sudo perf sched latency -s max`
- \* `sudo perf sched script`

# CPU性能分析

- \* `sudo perf record -F 99 -p 10805 -a --call-graph dwarf sleep 5`
- \* `sudo perf script | ./stackcollapse-perf.pl > out.perf-folded`
- \* `./flamegraph.pl out.perf-folded > dnd.svg`

# 获取函数时间分布

- \* `sudo stap -v -L 'process("/mnt/engine/fingerprint-y/fingerprint-y/bin/dnd").function("Register")'`
- \* `sudo stap -v histogram-user-time.stp 'process("/mnt/engine/fingerprint-y/fingerprint-y/bin/dnd").function("Register@/data/home/liaotianyi/workspace/git/fingerprint-y/dnd/src/DndHandler.cc")'`



# 获取函数时间分布

\* `sudo stap -v histogram-user-times.stp 'process("/mnt/engine/smart-list/smart-list/bin/smart-list-9090").function("Compare@/mnt/home/compile/makepkg/arch/be-smart-list/src/engine/*")' 10`

# 打印耗时堆栈信息

- \* `sudo stap -v user-stack-time.stp 'process("/mnt/engine/smart-list/smart-list/bin/smart-list-9090").function("NormalizeText@/mnt/home/compile/makepkg/arch/be-smart-list/src/engine/smart-list-base.cc")' 1000`
- \* 使用c++filt  
`_ZN10smart_list13SmartListBase13NormalizeTextERK  
SsPSt6vectorImSalmEE`

# 找出进程系统调用

- \* `sudo stap -v -x 27001 count-user-kernel.stp`
- \* 可能需要循环执行，找出可能的系统调用，再分析具体的系统调用

# 分析第三方库函数

- \* `sudo stap -v -e 'probe process("/opt/lib/libhiredis.so").function("redisGetReply") {print_ubacktrace();}'`
- \* `sudo stap -v histogram-user-time.stp 'process("/opt/lib/libhiredis.so").function("redisGetReply")' 5`

# 提供的分析能力

- \* 代码热点路径
- \* 函数耗时时间，执行次数统计
- \* 函数堆栈信息，函数的参数跟踪，函数本地变量
- \* 进程的系统调用分析
- \* 第三方库函数分析

# 参考

\* 脚本代码:

<git@code.aliyun.com:arch/systemtap-script.git>

\* System-tap:

<https://sourceware.org/systemtap/documentation.html>