程序优化工具—valgrind 入门

1. 简介

Valgrind is an instrumentation framework for building dynamic analysis tools. There are Valgrind tools that can automatically detect many memory management and threading bugs, and profile your programs in detail. You can also use Valgrind to build new tools.

官网： <http://valgrind.org/>

1. 安装

a. 去官方网站 <http://valgrind.org/downloads/current.html> 下载最新安装包，如 valgrind 3.13.0 (tar.bz2) 解压；

b. 常规源码编译安装：./autogen.sh； ./configure; make; make install;

c. 安装成功 valgrind --version 输出版本号；

1. **内存泄漏检测**

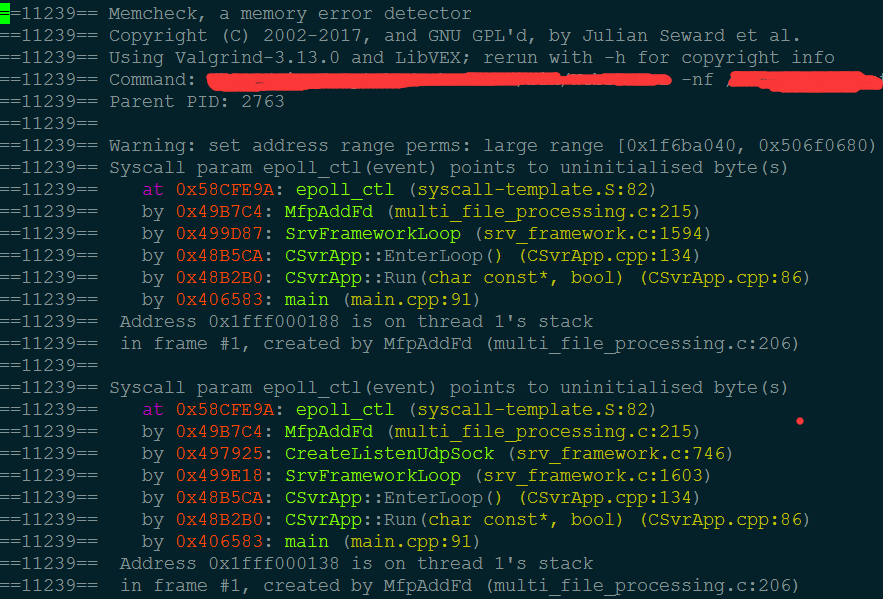
a. 正常编译目标程序 (如 g++ -g -o valgrind\_tst main.cpp)，官方推荐加上 -g 选项；

b. valgrind --tool=memcheck --leak-check=full --log- file=memchk.log valgrind\_tst [Your programme option];

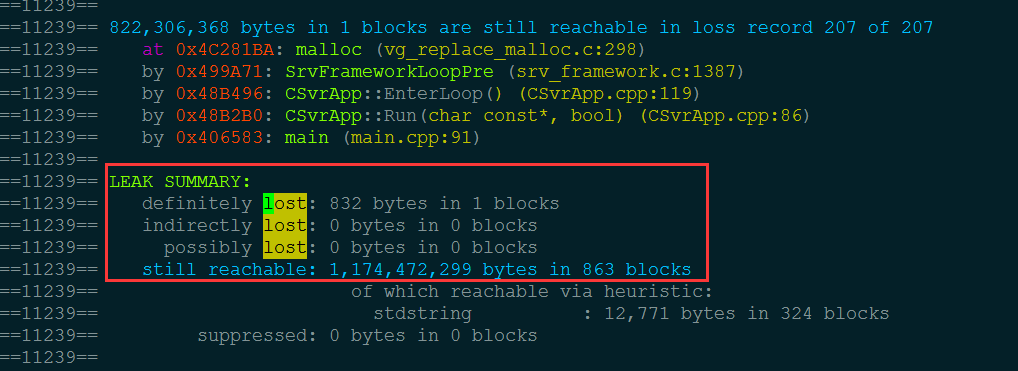
c. --tool=memcheck 表示使用工具集中的内存检测，其他选项参考 valgrind -h;

d. 特别的 --trace-children = yes 开启这个选项将使 valgrind 追踪到子进程，如果后台服务无法设置前台运行，应开启这个选项；

e. 结果输出，直接 vim memchk.log(由命令中的 --log-file 指定)查看：



结果中比较重要的是：



definitely lost: 确定有内存泄漏，表示在程序退出时，该内存无法回收，也没指针指向该内存(首地址)；

indirectly lost: 间接内存泄漏，比如结构体中定义的指针指向的内存无法回收；

possibly lost： 可能出现内存泄漏，比如程序退出时，没有指针指向一块内存的首地址了，但由其他某个指针能推算出首地址；

still reachable: 程序没主动释放内存，在退出时候该内存仍能访问到，比如全局 new 的对象没 delete，由于操作系统会回收，所以此类问题可忽略;

最严重的是 definitely lost 和 indirectly lost，检测结果文件中已给出了具体函数和源文件。

1. 调用耗时检测(常用于性能调优)

a. 正常编译目标程序同上；

b. valgrind --tool=callgrind -v Your\_Programme [Your\_Programe\_Option];

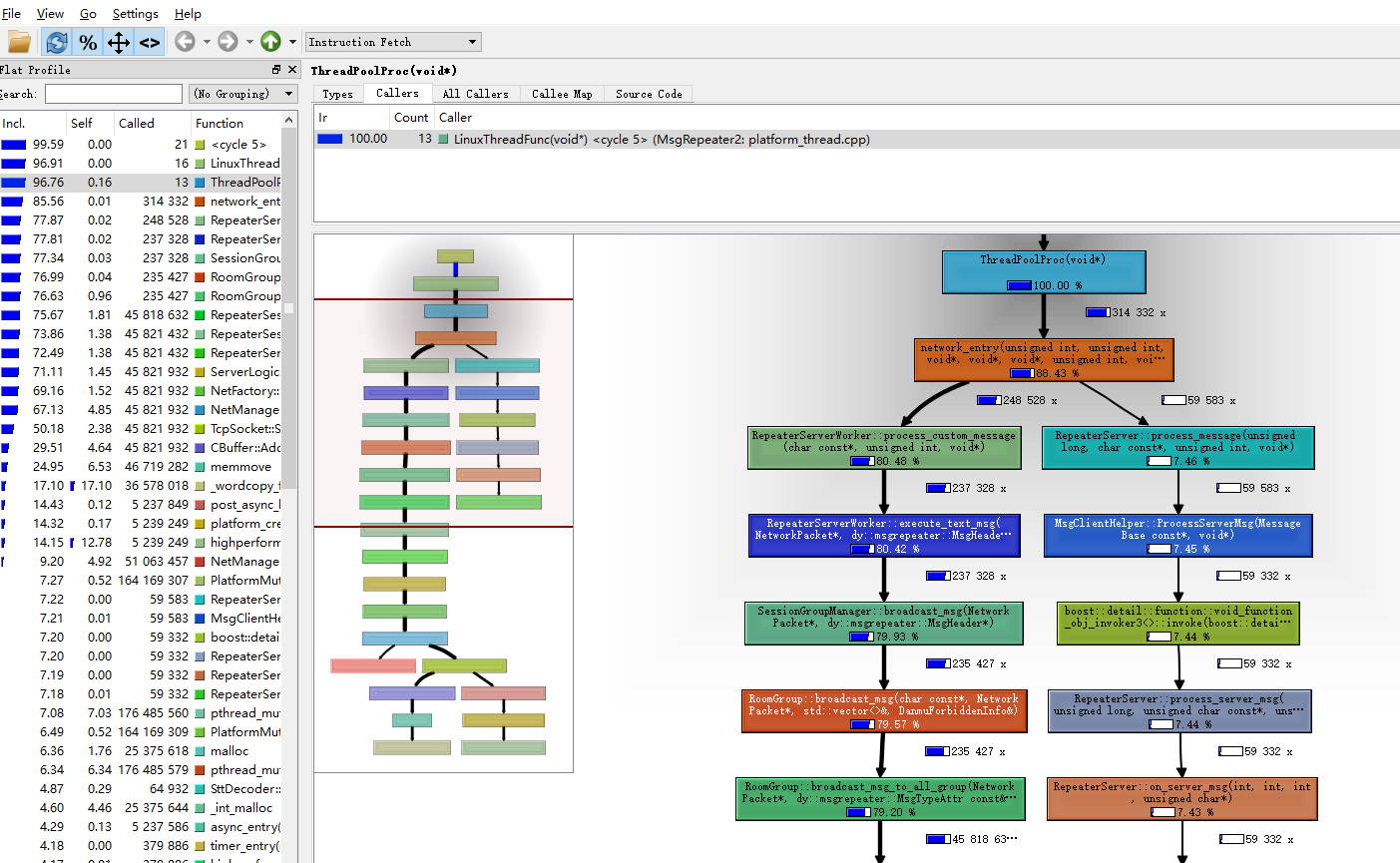
c. --tool=callgrind 表示使用调用检测工具，如果是后台服务，需要服务能正常退出，不要通过kill -9 的方式结束进程；

d. 由于 callgrind 的原理是时间点采样，所以被测程序最好在合适的压力下运行合适长的时间；

e. 如果需要查看各线程的性能数据，可使用--separate-threads=yes 选项;

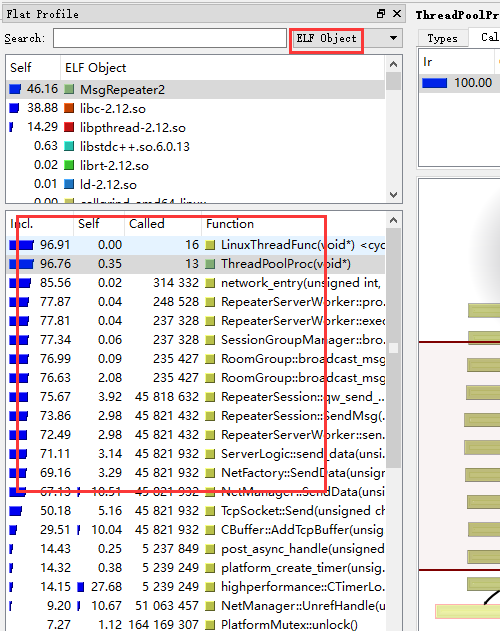
f. 查看结果，运行完后，将输出 callgrind.out.PID 文件，该文件可被 gprof2dot 等工具解析转化为 dot，再由 dot 转化为图片输出，也可以直接用 callgrind\_annotate 直接解析打印，但最方便的是使用图形工具解析，如 windows 上的 qcachegrind;

g. qcachegrind 安装略，以下是 qcachegrind 打开上述 callgrind.out.PID 文件的效果：

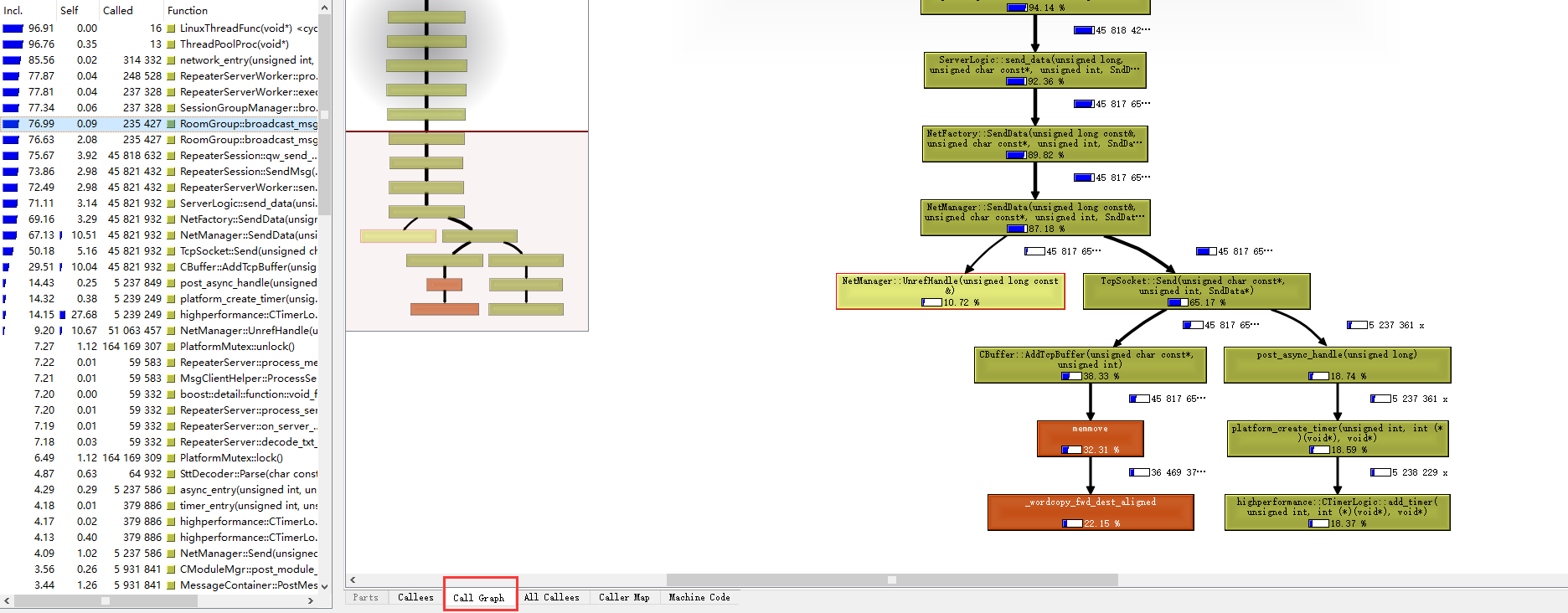


左边列出了调用parts、调用栈 cost 排序等，右边列出了所有调用者、被调用者以及调用图等。

h. 分析程序的性能损耗在哪，可以选择左侧 Flat File --> group by ELF object，找到感兴趣的 object 文件:



1. Self表示自身执行的耗时，Incl表示包含调用函数的总耗时，选择感兴趣的函数，在右边可查看该函数的调用关系图:



根据耗时分布找到程序的性能瓶颈，针对性的优化，关于 qcachegrind 的详细说明，可查阅官网 <http://kcachegrind.sourceforge.net/html/Home.html>

1. 其他

Valgrind 是一套强大的 profile 工具集，上述两个是经常用到的工具，但由于使用 valgrind 检测时性能急剧下降(至少10倍)，所以不要在线上直接 valgrind.

除 memcheck 和 callgrind 外，helgrind 是另一个非常有用的工具，它主要用于检测：

1)  资源不安全访问   
2) 死锁问题   
3)  POSIX pthreads API的错误使用等

这里不再叙述，可自行 google.