1. **概述**

Valgrind是一款用于内存调试、内存泄漏检测以及性能分析的软件开发工具。是一个运行时诊断工具，它可以监视一个指定程序的活动并打印出代码中可能存在的各种各样的内存管理问题。它用来检查程序的内存使用状况，尤其是对没有明显特征的内存越界、内存泄露等在测试中不能直观检查的情况，效果明显。

这款工具是运行在linux下的，可以用来定位c/c++程序中内存使用方面的错误。包括：内存泄漏、使用未初始化的内存、读/写已释放的内存、读/写内存越界、使用malloc/new/new[]和free/delete/delete[]不匹配，等等。

Valgrind的最初作者是Julian Seward，他于2006年由于在开发Valgrind上的工作获得了第二届Google-O'Reilly开源代码奖。[

到3.3.0版本为止，Valgrind支持x86、x86-64以及PowerPC上的Linux。除此之外，还有一些其它非正式支持的类Unix平台（如FreeBSD、NetBSD以及Mac OS X）。

valgrind官方网站：http://valgrind.org/，是一款open source软件。Valgrind遵守GNU通用公共许可证条款，是一款自由软件。

1. **安装**
   1. **可以直接到valgrind官网下载源码编译、安装，由于valgrind编译依赖很多其他库，直接下载安装会比较耗时。推荐使用Jumbo包管理工具安装。**
   2. Jumbo安装：直接在测试机上运行如下命令即可:

bash -c "$( curl http://jumbo.baidu.com/install\_jumbo.sh )"; source ~/.bashrc

* 1. 使用Jumbo搜索、安装valgrind，命令如下：

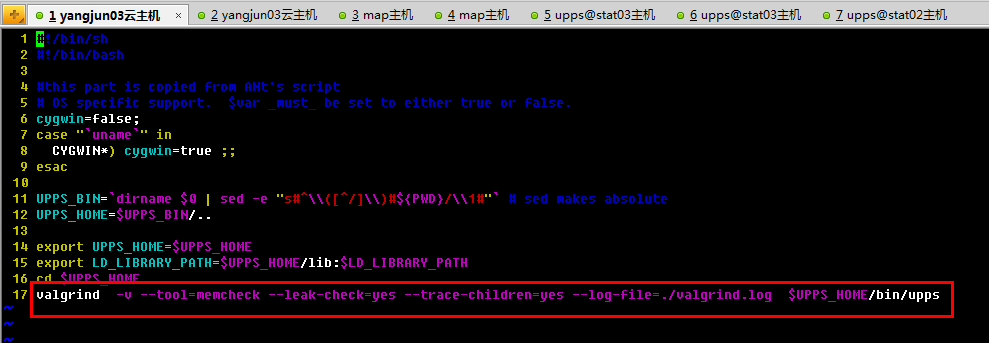
jumbo search valgrind

jumbo install valgrind

* 1. Valgrind安装完毕，可使用valgrind –version查看版本信息，如下图：



1. **使用**
   1. 使用valgrind启动被测系统，以upps为例，修改upps启动脚本，如下：



* 1. 用法: valgrind [options] prog-and-args [options]。参数解释如下：
* 常用选项，适用于所有Valgrind工具：

-tool=<name> 最常用的选项。运行 valgrind中名为toolname的工具。默认memcheck。

h –help 显示帮助信息。

-version 显示valgrind内核的版本，每个工具都有各自的版本。

q –quiet 安静地运行，只打印错误信息。

v –verbose 更详细的信息, 增加错误数统计。

-trace-children=no|yes 跟踪子线程? [no]

-track-fds=no|yes 跟踪打开的文件描述？[no]

-time-stamp=no|yes 增加时间戳到LOG信息? [no]

-log-fd=<number> 输出LOG到描述符文件 [2=stderr]

-log-file=<file> 将输出的信息写入到filename.PID的文件里，PID是运行程序的进行ID

-log-file-exactly=<file> 输出LOG信息到 file

-log-file-qualifier=<VAR> 取得环境变量的值来做为输出信息的文件名。 [none]

-log-socket=ipaddr:port 输出LOG到socket ，ipaddr:port

* LOG信息输出选项：

-xml=yes 将信息以xml格式输出，只有memcheck可用

-num-callers=<number> show <number> callers in stack traces

-error-limit=no|yes 如果太多错误，则停止显示新错误? [yes]

-error-exitcode=<number> 如果发现错误则返回错误代码 [0=disable]

-db-attach=no|yes 当出现错误，valgrind会自动启动调试器gdb。[no]

-db-command=<command> 启动调试器的命令行选项[gdb -nw %f %p]

* 适用于Memcheck工具的相关选项：

-leak-check=no|summary|full 要求对leak给出详细信息? [summary]

-leak-resolution=low|med|high how much bt merging in leak check [low]

-show-reachable=no|yes show reachable blocks in leak check? [no]

* 1. **启动程序后，对程序进行发送query，query的量根据程序本身的情况来确定，基本省覆盖所有回归功能的query，以及部分的随机query。以压力的形式发送到待测试的程序。在这个过程中valgrind会将程序走到路径的异常情况记录下来，并且给出详细的调用关系记录，以及出错的地方在什么地方。**

1. **日志分析**
   1. Valgrind记录文件的主要错误有以下几种：

-内存未初始化：Conditional jump or move depends on uninitialised value

-读写越界：Invalid read/write of size

-内存泄露：definitely lost

-原串与目标串内存交叠：Source and destination overlap in strcpy/memcpy

-非法释放或者删除：Invalid free() / delete / delete[]

-含义和5相同：Mismatched free() / delete / delete[]

1. **注意**
   1. valgrind本身有一些无法检查的错误：

**-Valgrind可以检查堆内存越界和内存泄露，可以检查简单的堆栈溢出且出core的问题，但是无法检查栈内存越界。**

-堆栈错误出现了core，所以valgrind可以检测到错误信息，但是由于堆栈已经被破坏，无法准确还原真实的函数调用栈。

-对于全局指针，进程在收到15信号后是不进行释放，但是对应的内存有可能被系统回收。

**-对于所有指针，进程在收到9信号后理解退出，valgrind无法决定是否有内存泄漏。不建议强制停止被测系统进程，否则根据系统退出之后valgrind日志不能准确判断是否存在内存泄漏。**

* 1. Valgrind需要有实际的数据运行到相应的分支，产生相应的问题，才能对问题进行记录，不论是读写越界，还是内存泄露，都必须有query走到。但是在测试程序中，测试者对于程序的某些异常分支不了解，或者了解也不能构造出有效的case命中，这些异常分支valgrind无法检测，需要使用其他的测试辅助手段。