同济大学计算机网络

实验报告



姓名： 涂远鹏-1652262

题目： 程序的静态及动态编译

**1.Linux 下的动态编译**

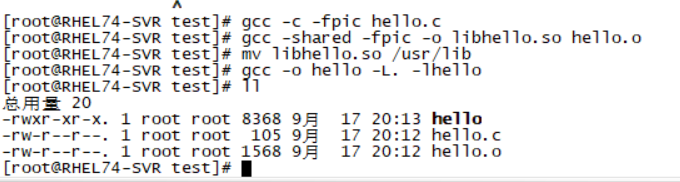
**(1)什么叫动态编译?**

答：动态编译是某些[程式](https://zh.wikipedia.org/wiki/%E7%A8%8B%E5%BC%8F%E8%A8%AD%E8%A8%88" \o "程式设计)[语言](https://zh.wikipedia.org/wiki/%E7%A8%8B%E5%BC%8F%E8%AA%9E%E8%A8%80" \o "编程语言)在执行时用来增进效能的方法。尽管这技术源于[Self](https://zh.wikipedia.org/wiki/Self" \o "Self)[[来源请求]](https://zh.wikipedia.org/wiki/Wikipedia:%E5%88%97%E6%98%8E%E6%9D%A5%E6%BA%90" \o "Wikipedia:列明来源)，但使用此技术最为人所知的是[Java](https://zh.wikipedia.org/wiki/Java" \o "Java)。此技术可以做到一些只在执行时才能完成的最佳化。使用动态编译的[执行环境](https://zh.wikipedia.org/wiki/%E6%89%A7%E8%A1%8C%E7%8E%AF%E5%A2%83" \o "执行环境)一开始执行速度较慢，之后，完成大部分的编译和再编译后，会执行得比非动态编译程式快很多。因为初始化时的效能延迟，动态编译不适用于一些情况。在许多实作中，一些可以在[编译时期](https://zh.wikipedia.org/wiki/%E7%B7%A8%E8%AD%AF%E6%99%82%E6%9C%9F" \o "编译时期)做的最佳化被延到[执行时期](https://zh.wikipedia.org/wiki/%E5%9F%B7%E8%A1%8C%E6%99%82%E6%9C%9F" \o "执行时期)才编译，导致不必要的效能降低。[即时编译](https://zh.wikipedia.org/wiki/%E5%8D%B3%E6%99%82%E7%B7%A8%E8%AD%AF" \o "即时编译)是一种动态编译的形式。

动态编译的[可执行文件](http://baike.baidu.com/view/159830.htm" \t "https://blog.csdn.net/wuan584974722/article/details/_blank)需要附带一个的[动态链接库](http://baike.baidu.com/view/887.htm" \t "https://blog.csdn.net/wuan584974722/article/details/_blank)，在执行时，需要调用其对应动态链接库中的命令。所以其优点一方面是缩小了执行文件本身的体积，另一方面是加快了编译速度，节省了[系统资源](http://baike.baidu.com/view/53557.htm" \t "https://blog.csdn.net/wuan584974722/article/details/_blank)。缺点一是哪怕是很简单的程序，只用到了链接库中的一两条命令，也需要附带一个相对庞大的链接库；二是如果其他计算机上没有安装对应的[运行库](http://baike.baidu.com/view/1032404.htm" \t "https://blog.csdn.net/wuan584974722/article/details/_blank)，则用动态编译的可执行文件就不能运行。

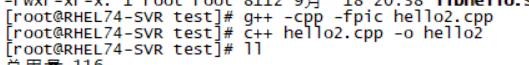
**(2)给出printf("hello，world");程序的gcc动态编译命令，可执行文件字节数是多少?**

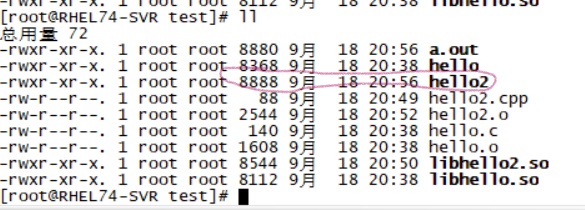
答：字节数为8368



**(3)给出 cout << "hello，world";程序的 c++/g++动态编译命令，可执行文件字节数是多少?**

答：字节数为8888



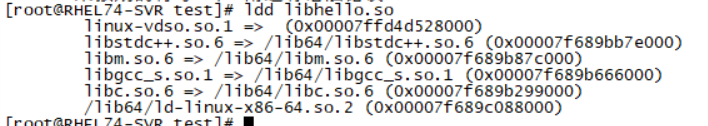


**(4)给出第一周作业中mysql\_demo.cpp的动态编译命令，可执行文件字节数是多少?**

2

**(5)如何查找某个可执行文件所依赖的动态链接库?**

答：可以通过ldd命令查看一个可执行程序依赖的共享库，如下图所示，通过ldd指令查看libhello.so所依赖的共享库：



CSDN上的详细解释：****

**2.Linux 下的 gcc 静态编译**

**(1)什么叫静态编译?**

答：通常情况下，对函数库的链接是放在编译时期（compile time）完成的。所有相关的对象文件（object file）与牵涉到的函数库（library）被链接合成一个可执行文件 （executable file）。程序 在运行 时，与函数库再无瓜葛，因为所有需要的函数已拷贝到自己门下。所以这些函数库被成为静态库（static libaray），通常文件 名为“libxxx.a”的形式。

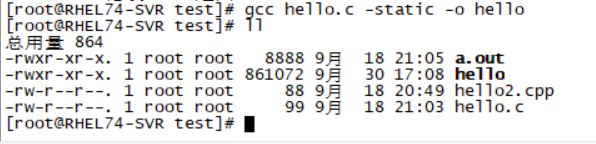
**(2)给出printf("hello，world");程序的gcc静态编译命令，可执行文件字节数是多少?**

答：

首先安装静态编译所需的包glibc-static-2.17-222.el7.x86\_64.rpm（从官网上下即可）



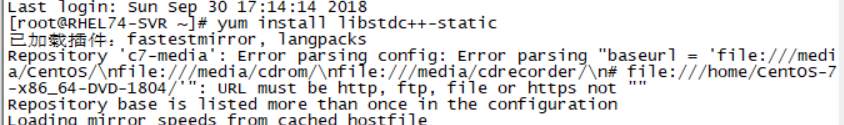
Gcc编译源程序生成的可执行文件的字节数为861072字节：



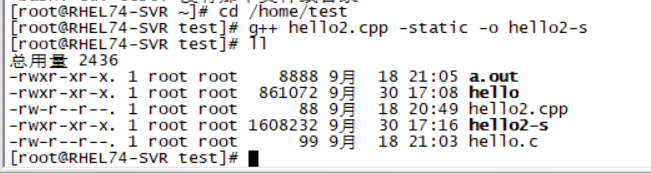
**3. Linux 下的 c++/g++静态编译**

答：

首先安装stdc++对应的包：

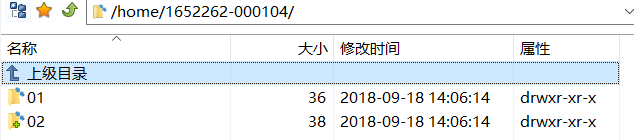


然后就是编译过程生成的可执行文件字节数为1608232，比之前的大多了：



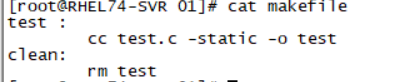
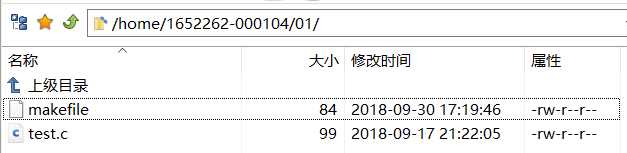
**4. 按要求写出下列几种常用情况的静态编译测试样例**

**4.1新建1652262-000104文件夹并新建对应的源程序：**

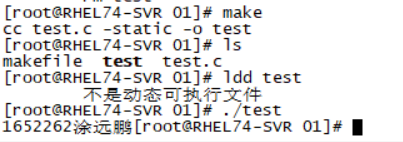


**4.2在子目录01下建立test.c，并写出满足要求的makefile文件：**

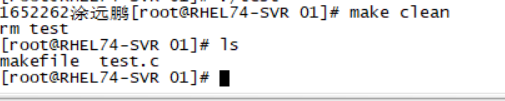
(1)编写对应的makefile内容显示如下：



(2)测试make指令效果，显示结果正确：



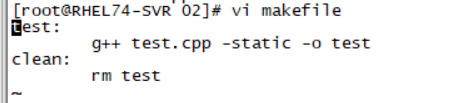
(3)测试make clean指令效果，显示结果正确：



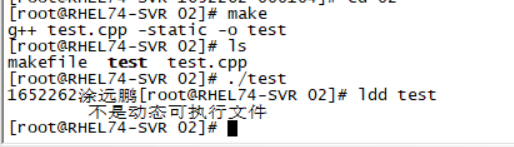
4.3**在子目录02下建立test.cpp，并写出满足要求的makefile文件：**

(1)创建文件夹并编写对应的makefile内容显示如下：

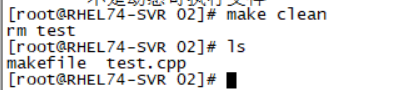




(2)测试make指令效果，显示结果正确：

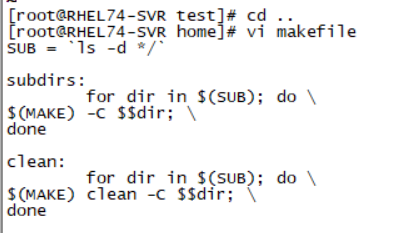


(3)测试make clean指令效果，显示结果正确：

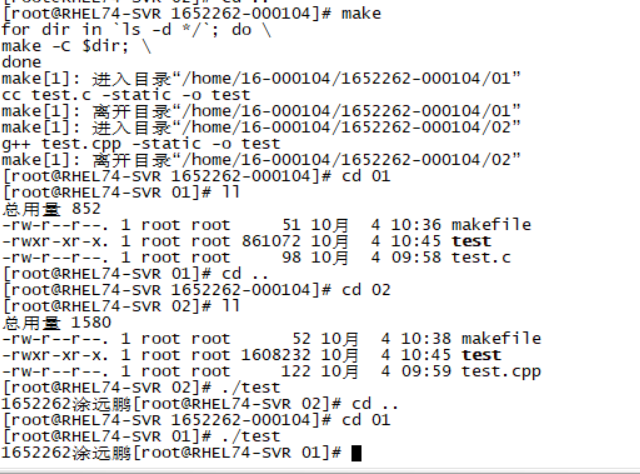


4.4**在1652262-000104目录下写一个满足要求的makefile文件：**

(1)makefile内容显示如下：



(2)测试make指令效果，显示结果正确：



(3)测试make clean指令效果，显示结果正确：

