

Unix动态库和静态库比较Linux认证考试 PDF转换可能丢失图片或格式，建议阅读原文

https://www.100test.com/kao_ti2020/644/2021_2022_Unix_E5_8A_A8_E6_80_81_c103_644919.htm 静态库和动态库的区别 文件预览 文件目录树如下，如你所见，非常简单。

```
1. libtest/ 2. |-- lt.c 3. |-- lt.h 4. `-- test.c #lt.c 1. 4. 5. #include 6. 7. void myprint(void) 8. { 9. printf("Linux library test!\n"). 10. } #lt.h 1. 4. 5. void myprint(void). #test.c 1. 4. 5. #include "lt.h" 6. 7. int main(void) 8. { 9. myprint(). 10. return 0. 11. }
```

先看静态库 首先做成静态库 liblt.a。

```
1. $ gcc -o lt.o -c lt.c 2. $ ar cqs liblt.a lt.o
```

再者，链接，

```
1. $ gcc test.o liblt.a -o test
```

这个时候再来看他的引用库情况。

```
1. $ ldd test 2. linux-gate.so.1 => /lib/libc.so.6 (0xb7e29000) 4. /lib/ld-linux.so.2 (0xb7f6e000)
```

动态库 做成动态库 liblt.so。

```
1. $ gcc -o lt.o -c lt.c 2. $ gcc -shared -Wall -fPIC -o liblt.so lt.o -shared
```

该选项指定生成动态连接库（让连接器生成T类型的导出符号表，有时候也生成弱连接W类型的导出符号），不用该标志外部程序无法连接。相当于一个可执行文件 -fPIC：表示编译为位置独立的代码，不用此选项的话编译后的代码是位置相关的所以动态载入时是通过代码拷贝的方式来满足不同进程的需要，而不能达到真正代码段共享的目的。 -L：表示要连接的库在当前目录中 -ltest：编译器查找动态连接库时有隐含的命名规则，即在给出的名字前面加上lib，后面加上.so来确定库的名称 LD_LIBRARY_PATH：这个环境变量指示动态连接器可以装载动态库的路径。链接方法I，拷贝到系统库里再链接，让gcc自己查找

```
1. $ sudo cp liblt.so /usr/lib 2. $ gcc -o test test.o -llt
```

这里我们可以看到了 -llt 选项，-l[lib_name] 指定库名

，他会主动搜索lib[lib_name].so。这个搜索的路径可以通过gcc --print-search-dirs来查找。链接方法II，手动指定库路径1. \$ cc -o test test.o -llt -B /path/to/lib这里的-B 选项就添加/path/to/lib 到gcc搜索的路径之中。这样链接没有问题但是方法II中手动链接好的程序在执行时候仍旧需要指定库路径（链接和执行是分开的）。需要添加系统变量LD_LIBRARY_PATH: 1. \$ export LD_LIBRARY_PATH=/path/to/lib 这个时候再来检测一下test程序的库链接状况(方法I情况) 1. \$ ldd test 2. linux-gate.so.1 => /usr/lib/liblt.so (0xb7f58000) 4. libc.so.6 => /lib/libc.so.6 (0xb7e28000) 5. /lib/ld-linux.so.2 (0xb7f6f000)恩，是不是比静态链接的程序多了一个liblt.so？恩，这就是静态与动态的最大区别，静态情况下，他把库直接加载到程序里，而在动态链接的时候，他只是保留接口，将动态库与程序代码独立。这样就可以提高代码的可复用度，和降低程序的耦合度。另外，运行时，要保证主程序能找到动态库，所以动态库一般发布到系统目录中，要么就在跟主程序相对很固定的路径里，这样不管主程序在本机何时何地跑，都能找得到动态库。而静态库只作用于链接时，运行主程序时，静态库文件没存在意义了。编辑特别推荐: linux面试题参考答案 最简便的MySQL数据库备份方法 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com